CONFIGURATION CONTROL OF A MOBILE DEXTEROUS ROBOT:
REAL–TIME IMPLEMENTATION AND EXPERIMENTATION

David Lim, Homayoun Seraji
Jet Propulsion Laboratory
California, Institute of Technology
Pasadena, CA 91109, USA

Abstract

This paper describes the design and implementation of a real-time control system with multiple modes of operation for a mobile dexterous manipulator. The manipulator under study is a cinematically redundant seven degree-of-freedom arm from Robotics Research Corporation, mounted on a one degree-of-freedom mot orized platform. The manipulator-plus-platform system has two degrees-of-redundancy for the task of hand placement and orientation. The redundancy resolution is achieved by accomplishing two additional tasks using *the configuration control technique. The system allows a choice of arm angle control or collision avoidance jor the seventh task,* and *platform placement or elbow angle control jor the eighth task. In addition, joint limit avoidance tasks are automatically invoked when any of the manipulator joints approach their limits. The system is robust to singularities, and provides the capability of assigning weighting factors to end-effector, joint limit avoidance, and redundancy resolution tasks. The motion control algorithms are executed at* 1.1 *msec on two MC68040 processors in a VME-bus environment running the VxWorks real-time operating system, The paper describes the hardware and software components of the VME environment. Experimental results on real-time control of the Robotics Research arm are also presented in the* paper.

1 Introduction

In October 1990, the United States National Aeronautics and Space Administration (NASA) initiated a research and development project on supervised telerobotic inspection [5] at the Jet Propulsion laboratory (J Pi,). The goal of this project is to develop the necessary technologies for the inspection of space structures, such as the Space Station, using remote robots for sensor placement, under supervisory control. The purpose of the inspection is to monitor the health and assess possible damage to the structure by employing a number' of sensing devices as deemed appropriate for the task, An essential component, of a remote inspection system is a mobile dexterous robotic manipulator for sensor placement and the associated control system. 'l'his paper describes the design and implementation of a multi- mode real- time control system for a mobile dexterous manipulator used in the NASA supervised inspection project.

The paper is structured as follows. In Section 2, the hardware used by the manipulator control system is described. The algorithms used in the control system are discussed in Section 3. *In* Section 4, the graphical user interface is outlined. In Section 5, we present the software architecture of the VM E environment used for real- time control of the manipulator. Experimental results on real--time control of the dexterous manipulator are presented in Section 6. Section 7 draws some conclusions from this work, and discusses the directions of future research and development.

2 Hardware System Description

In this section, we describe the hardware of the Manipulator Control System (MCS). The hardware structure is shown in Figure 1 and consists of a Robotics Research Corporation's model K1 207 seven degree of- freedom (DOF) arm/control unit, a VME-based chassis with two MC68040 processor boards and additional interface cards, two joysticks, a motorized platform /control unit on which the arm is mounted, and a Silicon Graphics IRIS workstation.

The system is divided into the local and remote subsystems. In our current hardware manifestation of this architecture, the local computer hardware is the IRIS workstation, and the remote hardware is the VME-based real-time system which controls the 7- DOF arm and the one DOF platform, Sensor information is received from an integrated sensor/end--effector (ISEE) unit consisting of two CCD cameras with controlled lights, two infrared triangulation based proximity sensors, a gas sensor, a temperature sensor, a force/torque sensor, and a gripper. Feedback data from the arm control unit for joint positions, joint velocities, and torques is also available. The system architecture contains an explicit division between the local and remote parts of the system. This is needed to address the problems of at-a-distance inspection of orbital platforms, The robotic inspection laboratory setup is shown in Figure 2. 'l'his figure shows the remote site where the inspection task is performed and consists of the arm with the ISEE, the platform, and a one-third scale mockup of part of the Space Station truss structure. The local site consists of an operator control station where the operator resides and is referred to as the '(cupola". Within the cupola are the IRIS workstation, two color monitors, a stereo color monitor, and two joysticks.

The dexterous manipulator used in this study is cinematically redundant with seven revolute joints in an alternating roll/pitch sequence beginning with the shoulder roll at the base and ending with the tool- plate roll at the hand. The shoulder has both a roll and a pitch DOF, the elbow has an extra roll DOF along the upper-arm in addition to the conventional pitch between the upper-arm and forearm, and the wrist has a roll DOF along the forearm, a pitch between the forearm and hand, and a roll about the tool-plate. 'l'he upper-arln roll motion allows the arm plane (formed by the upper-arm and forearm) to rotate, thus providing the capability for arm configuration control. The arm pedestal is mounted on a mobile platform of a motorized rail which provides one additional translational degree- of- freedom that can be treated as a prismatic joint. Therefore, the complete manipulator system has eight independent joint degrees- of--freedom. This system has two degrees-of- redundancy, i.e. two '(extra" joints, since six joints arc sufficient for the basic task of hand position and orientation in the three- dimensional workspace.

The Robotics Research arm is controlled by a real-time microprocessor- based controller developed at JPL that uses the configuration control algorithm's [9, 10, 11] for high- level dexterous motion control that interfaces directly with the Multibus-based arm control unit supplied by the manufacturer. The real- time controller is a VMEbus- based system that uses two Motorola MC68040 processors along with various data acquisition, memory, and communication boards. The VME controller is linked via socket communication to the Silicon Graphics IRIS workstation, which serves as the host computer for the graphical user interface. The controller also bas a shared memory interface allowing a high speed communication link with other systems. A separate image processing VM E chassis currently uses this interface to monitor the Cartesian position of the end- effector. The real-time VME chassis and' the arm control unit Multibus chassis are connected via a two- card VME-to-Multibus adaptor set from the BIT3 Corporation. 'l'his allows a high speed bi- directional shared memory interface between the two buses. The reason for this design choice is to have no software development on the Multibus system. Thus, the control system on the VME chassis treats the arm control unit as a joint space position controller. 'l'he control architecture simplifies the integration of future generations of higher-performance hardware and new control techniques as they become available, and thus provides a growth capability that extends the technical life of the arm control system.

The robot is mounted on a platform which is motorized. The commands to the platform control unit (built by Compumotor Division, Parker Hannifin Corporation) arc sent through a serial port. The platform control unit is capable of providing very accurate position control (0.01 2 mm accuracy). However, due to the serial port link, communication with the platform control unit occurs at a slow rate relative to the arm control unit.

3 Control System Description

The manipulator Cartesian control flow diagram is shown in Figure 3. The configuration control technique [9, 10, 11] developed at JPL is implemented in the VME environment for the seven DOF arm plus the one DOF mobile platform. This technique allows specification of additional tasks for redundancy resolution. Currently, two choices for the seventh task arc available: arm angle control for elbow placement, or obstacle avoidance to reach through an opening. For the eighth task, two choices are available: platform position control, or elbow angle control. ‘l’he major software modules for the control system are the forward kinematics and Jacobian computations, a singularity-robust inverse kinematic computation, and a real-time trajectory generation routine.

The computations of the forward kinematics and Jacobian of the 7-DOF manipulator utilize Craig's interpretation of Denavit-Hartenberg (DH) parameters for frame assignment [2, 6]. This method provides direct computation of the manipulator Jacobian in the world frame of the robot. In addition, the forward kinematics and Jacobian arc also computed for the obstacle avoidance task [3, 4]. A singularity- robust inverse kinematics algorithm is implemented. This technique is known as the damped-least-squares (DLS) method [7, 11, 13]. Basically, the method relies on weighting large joint velocities against large task-space errors. The resultant computation of the joint velocities has the following form [11]:

$$\dot{\theta}_d = \left[J^T W_t J + W_v\right]^{-1} J^T W_t \left[\dot{X}_d + K E\right] \tag{1}$$

where W_t and W_v are the task- space error weights and joint velocity damping weights, *J is* the augmented Jacobian matrix, X_d and X arc the desired and actual configuration vectors, $E = X_d$ — X is the DLS error term, and K is a diagonal matrix with positive elements that represent erroı feedback gains. Note that (1) can also be written as:

$$\dot{\theta}_d = \left[J_e^T W_e J_e + J_c^T W_c J_c + W_v\right]^{-1} \left[J_e^T W_e \dot{Y}_d + K_e E_e + J_c^T W_c \dot{Z}_d + K_c E_c\right] \tag{2}$$

where the subscript e refers to the basic task of positioning the hand, and the subscript c refers to additional tasks for redundancy resolution. Cholesky decomposition is used to find $\dot{\theta}_d$. It can be seen that as the Jacobian becomes singular, the velocity weight dominates in the inverse matrix term in (2), reducing the commanded joint velocities. ‘l)hc reduced joint velocities, in turn, act to retard the arm from reaching the singular configuration.

In the configuration control implementation, the "arm angle" is defined as the angle between the arm plane SEW and the vertical reference plane passing through the line *SW,* where S, E and *W* refer to the origins of the shoulder, elbow and wrist frames, respectively [6], This angle uniquely specifies the elbow position for a given hand frame, and together with the hand coordinates gives a complete representation of the geometric posture of the whole arm in almost the entire workspace. In the control software, we use a simple and efficient method described in [6] for computing the arm angle and the associated constraint Jacobian. The "elbow angle" is defined as the angle between the upper- arm SE and the forearm EW, [10]. The platform position is defined to be the position of the base of the robot with respect to a given world frame. The obstacle avoidance task allows the arm to reach through an opening while using the redundancy to avoid collision *[3, 4]*.

3.1 DLS Error Computation

Since the DLS error term in (1) is defined as $E = X_d - X$, for the three Cartesian hand positions, the DLS error is the arithmetic difference of the three individual scalar components. Similarly for the arm angle, ψ, the DLS error is the arithmetic difference between the measured and desired arm angles. However, some care should be taken since the arm angle is cyclic. In the experiments, the arm angle is defined to lie between ±180°, thus the error in arm angle must be within ±180°, and the MCS software checks for this condition. The hand orientation error is slightly more involved. ‘l’he difference R_{diff} between the desired and measured orientations, R_d and R_m, can be expressed as follows [2]:

$$R_{diff} = R_d R_m^{-1} = R_d R_m^T \tag{3}$$

where R is the 3 x 3 rotation matrix. ‘1’0 obtain a three component Version of R_{diff}, the equivalent angle-axis form of R_{diff} needs to be computed [2]. Let $\hat{k}$ be the equivalent axis unit vector and ϕ be the angular rotation about $\hat{k}$. Also, let $\hat{n}, \hat{o}$, and $\hat{a}$ be the columns of the rotation matrix R. Then from [1, 2]:

$$\cos\phi = \frac{\hat{n}_d \cdot \hat{n}_m + \hat{o}_d \cdot \hat{o}_m + \hat{a}_d \cdot \hat{a}_m - 1}{2} \tag{4}$$

$$\hat{k}\sin\phi = \frac{\hat{n}_d \times \hat{n}_m + \hat{o}_d \times \hat{o}_m + \hat{a}_d \times \hat{a}_m}{2} \tag{5}$$

From equations (4) and (5), $\hat{k}$ and ϕ can be computed. The norm of equation (5) yields sin ϕ, dividing this by (4) gives $atan2(\phi)$ and thus $\phi.\hat{k}$ can then be solved for by substituting ϕ back into (5). The equivalent angle--axis form of R_{diff} is simply $\hat{k}\phi$. However, since we are calculating an error term, $\hat{k}$ sin ϕ can be used as an approximation for $\hat{k}\phi$, since sin@ $\approx \phi$ when $\phi \approx$ O. Thus (5) can be used as an approximation for the orientation error, and is much simpler computationally.

3.2 Kinematic Analysis of Platform Motion

In this section, we present the kinematic analysis of the arm-plus-platform system. In order to simplify the analysis, wc disregard the three minor joints at the wrist and consider the four major joints of the RRC arm; namely the shoulder roll and pitch joints θ_1, θ_2, and the elbow roll and pitch joints θ_3, θ_4 as shown in Figure 4. The base platform motion on the track is along tile x-axis of the world frame and is treated as the prismatic joint θ_5. Let the task variables of interest be the wrist Cartesian coordinates $\{x, y, z\}$ and the arm angle ψ. The forward kinematic model relating $\{x, y, z, \psi\}$ to $\{\theta_1, \theta_2, \theta_3, \theta_4, \theta_5\}$ has been found to be [6]

$$
\begin{aligned}
x &= c_1 s_2 + c_1 s_2 c_4 + s_4(c_1 c_2 c_3 - s_1 s_3) + \theta_5 \\
y &= s_1 s_2 + s_1 s_2 c_4 + s_4(s_1 c_2 c_3 + c_1 s_3) \\
z &= c_2 + c_2 c_4 - s_2 c_3 s_4 + d \\
\psi &= atan2\{s_2 s_3 s_4, \frac{s_4}{r}[c_2 s_4 + s_2 c_3(1 + c_4)]\} = atan2\{u, v\}
\end{aligned}
\tag{6}
$$

where u and v arc the arguments of the $atan2$ function, $r = (2 + 2c_4)^{1/2}$ denotes the reach of the arm (i.e., the shoulder-wrist distance SW), h is the shoulder height, the upper-arm and forearm lengths are taken to be unity $(SE = EW = 1)$, and $s_i = \sin \theta_i, c_i = \cos \theta_i$. in deriving equation (6), the small offsets at the shoulder and elbow joints of the arm are ignored relative to the link lengths so that the analysis is mathematically tractable.

Since the robot system shown in Figure 4 has five independent joint degrees-of-freedom, we can control another task variable in addition to $\{x, y, z, \psi\}$. In this study, the additional task variable is chosen to be the "elbow angle" ϕ, which is related to the joint angles by

$$\phi = 180° + \theta_4 \tag{7}$$

and determines the reach of the arm. From triangle WEF, *we* obtain

$$r = SW = 2\ell \sin \phi/2 = 2 \sin \phi/2 \tag{8}$$

Hence the arm reach r is a simple sinusoidal function of the elbow angle ϕ, and ϕ can be used to control r directly. This equation can also be obtained by applying the cosine law to the *SEW* triangle to obtain $r = [2 + 2cos\theta_4]^{1/2}$ which can be reduced to equation (8) using the half-angle cosine formula. Notice that the arm angle ψ and the elbow angle ϕ represent two *independent* configuration parameters for the arm. The radius of the circle traversed by the elbow when the arm is executing a self--motion is a function of the elbow angle as $EF = \ell \cos \phi/2$. The variation of the arm reach r as a function of the elbow angle ϕ is shown in Figure 5. It is seen that when ϕ changes in the range {O, 1800}, r varies from O to 2; with r = O at ϕ = O (arm fully folded) and r = 2 at ϕ = 180° (arm fully extended).

Equations (6)-(7) represent the augmented forward kinematic model of the mobile robot system. 'l'he augmented differential kinematic model relating the joint velocities $\{\dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3, \dot{\theta}_4, \dot{\theta}_5\}$ to the resulting task velocities $\{\dot{x}, \dot{y}, \dot{z}, \dot{\psi}, \dot{\phi}\}$ is obtained by differentiating (6)-(7) as

$$\dot{X} = \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\psi} \\ \dot{\phi} \end{pmatrix} = \left[\begin{array}{cccc:c} J_{11} & J_{12} & J_{13} & J_{14} & 1 \\ J_{21} & J_{22} & J_{23} & J_{24} & 0 \\ 0 & J_{32} & J_{33} & J_{34} & 0 \\ 0 & J_{42} & J_{43} & J_{44} & 0 \\ \hdashline 0 & 0 & 0 & 1 & 0 \end{array} \right] \left| \begin{array}{c} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \\ \dot{\theta}_5 \end{array} \right| = J(\theta) \left| \begin{array}{c} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \\ \dot{\theta}_5 \end{array} \right| \qquad (9)$$

Because of the particular structure of the 5x5 augmented Jacobian matrix J, the expression) for $\det[J]$ simplifies considerably to

$$\det[J] = -J_{21}\left[J_{33}J_{42} - J_{32}J_{43}\right] \qquad (1o)$$

The elements of J that appear in (1 O) can be obtained from (6) as

$$\begin{aligned} J_{21} &= \frac{\partial y}{\partial \theta_1} = c_1 s_2 + c_1 s_2 c_4 + c_1 c_2 c_3 s_4 - s_1 s_3 s_4 = x - \theta_5 \\ J_{32} &= \frac{\partial z}{\partial \theta_2} = -s_2 - s_2 c_4 - c_2 c_3 s_4 = -PQ \\ J_{33} &= \frac{\partial z}{\partial \theta_3} = s_2 s_3 s_4 \\ J_{42} &= \frac{\partial \psi}{\partial \theta_2} = \frac{\partial \psi}{\partial u} \cdot \frac{\partial u}{\partial \theta_2} + \frac{\partial \psi}{\partial v} \cdot \frac{\partial v}{\partial \theta_2} = \frac{1}{u^2 + v^2}\left[v \frac{\partial v}{\partial \theta_2} - u \frac{\partial u}{\partial \theta_2}\right] \\ J_{43} &= \frac{\partial \psi}{\partial \theta_3} = \frac{\partial \psi}{\partial u} \cdot \frac{\partial u}{\partial \theta_3} + \frac{\partial \psi}{\partial v} \cdot \frac{\partial v}{\partial \theta_3} = \frac{1}{u^2 + v^2}\left[v \frac{\partial v}{\partial \theta_3} - u \frac{\partial u}{\partial \theta_3}\right] \end{aligned}$$

where $PQ = (x^2 + y^2)^{1/2}$ is the distance between the wrist projection on the x – y plane P and the robot base Q, and the partial derivatives in the above expressions are given by

$$\frac{\partial u}{\partial \theta_2} = c_2 s_3 s_4 \; ; \quad \frac{\partial v}{\partial \theta_2} = \frac{s_4}{r}\left[-s_2 s_4 + c_2 c_3 (1 + c_4)\right]$$

$$\frac{\partial u}{\partial \theta_3} = s_2 c_3 s_4 \; ; \quad \frac{\partial v}{\partial \theta_3} = \frac{s_4}{r}\left[-s_2 s_3 (1 + c_4)\right]$$

Substituting these expressions into (10) and simplifying the result yields the surprisingly simple expression

$$\mathrm{dct}[J] = r s_2(\theta_5 - x) \qquad (11)$$

This analysis shows that the arl~l-pills-~jlatforlrl system has the following singular configurations:

I : $\theta_5 - x = 0 \rightarrow x = \theta_5$ wrist and platform have the same r-coordinate
11: $s_2 = o$ $\rightarrow \theta_2 = 0°, 180”$ upper-arm is vertical
]]]: $r = o$ $\rightarrow \theta_4 = 180°$ arm is fully folded

in the singular configuration I (i.e., $x = \theta_5$), the first and fifth columns of J are multiples and hence θ_1 and θ_5 have identical effects on the task variables. In the singular configurations 11 and III (i. e., $s_2 =$ O and $c_4 = -1$), the 2x2 submatrix $\begin{pmatrix} J_{32} & J_{33} \\ J_{42} & J_{43} \end{pmatrix}$ of J becomes rank-deficient, and hence the joint angles $\{0_2, \theta_3\}$ do not affect the task variables $\{z, \psi\}$ independently.

Suppose that the motion trajectories $X_d(t) = [x_d(t), y_d(t), z_d(t), \psi_d(t), \phi_d(t)]^T$ are specified for the task variables. Then the required joint motions can be obtained by finding the closed-loop damped-least-squares solution of

$$\dot{X}_d = \begin{pmatrix} \dot{x}_d \\ \dot{y}_d \\ \dot{z}_d \\ \dot{\psi}_d \\ \dot{\phi}_d \end{pmatrix} = \begin{pmatrix} J_{11} & J_{12} & J_{13} & J_{14} & 1 \\ J_{21} & J_{22} & J_{23} & J_{24} & 0 \\ 0 & J_{32} & J_{33} & J_{34} & 0 \\ 0 & J_{42} & J_{43} & J_{44} & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix} \left| \begin{matrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \\ \dot{\theta}_5 \end{matrix} \right| = J\dot{\theta} \qquad (12)$$

that minimizes the cost function $L = ||\dot{X}_d - J\dot{\theta}||^2_{W_t} + ||\dot{\theta}||^2_{W_v}$ as

$$\dot{\theta}_d = [J^T W_t J + W_v]^{-1} J^T W_t [\dot{X}_d + K(X_d - X)] \qquad (13)$$

Typically, the pitch angle θ_4 can vary in the range $-180° \leq \theta_4 \leq 0°$ and hence the range of variation of the elbow angle ϕ is 0° to 180°, The most desirable elbow angle is $\phi = 90°$, which corresponds to the pitch angle θ_4 in mid-range and ensures that the arm is not in an over-stretched ($\phi \approx 180°$) or an under-stretched ($\phi \approx 0°$) configuration. The elbow condition $\phi = 90°$ can also be derived from another point of view. For the robot arm shown in Figure 4, the upper-arm SE and forearm EW define the arm plane A. The robot can be viewed as a two-link planar arm with joint rotations θ_2 and 0_4 which move the arm in the plane A. The arm plane A can rotate about the shoulder roll axis by θ_1 and about the upper-arm by θ_3. When the robot base is stationary ($0_5 =$ O), the wrist attains *maximum manipulability* when $\theta_2 - \theta_4 = 90°$, which is the classical two-link arm result [14]. Hence ensuring that the elbow angle $\phi = 90°$ guarantees the optimality of the wrist manipulability in the arm plane A when $\theta_5 =$ O.

Having established the desirability of the $\phi = 90°$ condition based on the above arguments, the platform can be positioned continuously to attain the target elbow angle $\phi_d = 90°$ while the wrist is executing the specified motion. Since the platform motion is often considerably slower than the arm movement, it is preferrable *not* to move the platform continuously. To this end, instead of tracking the constraint $\phi = \phi_d = 90°$ accurately, we can impose the inequality constraint

$$90° - \delta \leq \phi \leq 90° + \delta \tag{14}$$

where δ is a user-specified tolerance or margin. When the elbow angle ϕ is within the allowable bounds, the task weighting factor for ϕ is set to zero, and in this case base mobility will not be activated [unless the target, wrist position is otherwise unattainable]. When ϕ is outside these bounds, i.e. $\phi > 90° + \delta$ (arm over-stretched) or $\phi < 90° - \delta$ (arm under-stretched), the task weighting for ϕ changes smoothly to one as shown in Figure 6 and the platform is moved automatically to restore the optimal configuration $\phi = 90°$, without perturbing the wrist position. Thus the automatic motion of the base platform prevents undesirable over-stretched or under-stretched arm configurations, while enabling the wrist to reach positions in the workspace that would otherwise be unattainable.

Now let (x, y, z) represent the coordinates of the wrist and θ_5 be the x-coordinate of the base. The shoulder-wrist distance SW is given by

$$SW^2 = r^2 = 4\sin^2\phi/2 = (x - \theta_5)^2 + y^2 + (z - h)^2 \tag{15}$$

For a given wrist position W, the elbow angle ϕ is determined solely by the base location θ_5. To attain a desired elbow angle $\hat{\phi}$, the required base location is found from *(15)* as

$$\hat{\theta}_5 = x \pm w \tag{16}$$

where $w^2 = 4\sin^2\hat{\phi}/2 - y^2 - (z - h)^2$. Equation (16) gives two solutions for the platform position $\hat{\theta}_5$, given the desired elbow angle $\hat{\phi}$. These solutions are symmetrical about the line perpendicular from W onto the x-axis. Because of the slow communication rate with the platform control unit, the arm-plus-platform control system is not implemented as an integrated 5 DOF system. Instead, a "4 + 1" DOF approach is adopted whereby motion commands for the platform position θ_5 are computed based on the arm configuration $\{\theta_1 \ldots \theta_4\}$ using a stand-alone software, and are communicated through a serial port to the platform control unit for execution.

3.3 Joint Limit Avoidance

In addition to the eight tasks described above, an extra task is added for each manipulator joint that is near its limit. This is accomplished within the framework of the configuration control scheme [11]. When joint limits are approached, the system actually becomes "deficient" (as opposed to being "redundant"). The damped- least- squares algorithm automatically relaxes certain tasks based on their weighting factors. 'l'he joint limit avoidance task is formulated as an inequality constraint that is activated only when the joint is within its “soft” limit, and is inactive otherwise. interestingly, the formulation of the extra task is extremely simple. Observe that $J_\zeta^T W_\zeta J_\zeta = W_\zeta$ and that $J_\zeta^T W_\zeta$ reduces to W_ζ, where ζ indicates the joint limit avoidance task. Thus computationally the joint limit avoidance task is extremely fast. To avoid chattering when tile joint limit avoidance task is activated and deactivated, W_ζ is formulated as a continuous function of θ, e.g. at the lower joint limit:

$$W_\zeta = \begin{cases} 0 & \theta \geq \theta_{soft} \\ \frac{W_{max}}{2}\left[1-\cos\left(\pi\frac{\theta-\theta_{soft}}{\theta_{hard}-\theta_{soft}}\right)\right] & \theta_{hard} < \theta < \theta_{soft} \\ W_{max} & \theta \leq \theta_{hard} \end{cases} \tag{17}$$

where θ_{soft} and θ_{hard} are the user-defined soft and hard joint limits, A typical plot of equation (17) is shown in Figure 7, where the abscissa denotes the physical range of the joint angle. Equation (17) is applied at the lower joint limit, and a similar equation is applied at the upper joint limit. Note that the task weight is’ zero when the joint is within the user-specified soft joint limits. Also note that the task weight is a smooth continuous function of the joint angle to avoid chattering as the joint transitions in and out of joint limits.

3.4 Trajectory Generation

Two independent trajectory generators are implemented in the system. The first trajectory generator produces smooth continuous cycloidal functions to make the transition from the initial position/orientation to the final position/orientation in the specified time. A second via-point blending trajectory generator is also implemented in the system [1 2]. The via--point blending trajectory generator allows the specification of several via- points. The control system generates a continuous trajectory between the points, while smoothly blending the velocities from one via-point to the next.

3.5 Simulation mode

The control system provides an arm simulation mode in addition to real arm execution mode. During the real arm execution mode, the control system sends the measured joint angles to the IRIS. This ensures that the user views the actual arm configuration since the measured joint angles are used for graphic simulation. In the simulation mode, the control system simply outputs the joint setpoints to a different shared memory location than that used in the execution mode, and the commanded joint setpoints are not sent to the arm control unit, The control system transmits the commanded joint setpoints instead of the actual joint angles to the IRIS. The implementation of the simulation mode in the control system assures that the simulation will effectively duplicate real arm execution since the *same* code is executed in both cases.

4 Graphical User Interface

The graphical user interface enables the user to specify the command and control modes of operation, set control system parameters, and determine the manipulator stat, us. Figure 8 shows the IRIS windows for manipulator control. The user first selects the command mode by clicking the mouse on the appropriate button on the menu. The manipulator control system has three *command mode* options:

- *Teleoperated Command Mode*
- *Automated Command Mode*
- *Shared Command Mode*

In the teleoperated command mode, the user employs two industrial joysticks to generate the commanded velocity inputs to the manipulator system. These joysticks are built by Measurement Systems, inc. and are identical to the ones used by astronauts to operate the Remote Manipulator System (RMS) from the Space Shuttle bay. The first joystick has a 3-axis square handle and is used solely to command translation; the second joystick has a 3- axis rotational grip handle and is used for commanding orientation. The second joystick also has three mounted switches. The trigger is used to change the arm angle in one direction at a constant speed. The slide switch is used to move the platform in one direction at a constant speed. The momentary push-button switch has dual usage; it is used to change the direction of both the platform and the arm angle commands. 'I)he two joysticks can alternatively be used to send motion commands directly to the seven joint angles of the arm. The user can select the gains that map the joystick deflections into the arm displacements. Pre-stored values for LOW, MEDIUM, and HIGH gain settings can be selected, or the user can input the desired numerical values of the gains. In the automated command mode, the motion commands to the arm are issued by a trajectory generator software in the VME chassis. In this case, the user inputs on the keyboard or the slider bars the desired final values of the hand coordinates and arm angle or the target values of the joint angles, as well as the motion duration.

The system also provides shared command mode by combining the teleoperated and automated modes, where the commanded values for the arm coordinates are read both from the joystick channel and tile trajectory generator channel and added together to form the commanded arm coordinates. 'l'he shared command mode of operation is particular useful in applications where the hand is moved in automated mode by the trajectory generator software while the user is commanding the elbow motion through the arm angle using the joystick in teleoperated mode.

The user can also select any of the following *control modes* to operate the arm by clicking the mouse on the appropriate button on the IRIS screen:

- *Joint Control Mode*: Commands are issued to the seven joint angles of the arm and the platform.

- *Cartesian-World Control Mode:* Commands are expressed relative to a fixed user-defined frame of reference (the world frame).
- *Cartesian- World Relative Control Mode:* Motion commands are in the world frame coordinates measured relative to the current world frame coordinates of the hand.
- *Cartesian- Tool Control Mode:* Motion commands are in the end- effector coordinates measured relative to a reference frame displaced by the tool length from the current hand frame. The tool length is defined by the user from the graphical user interface.

The operator can choose between arm simulation mode or real arm execution mode with the click of a button. The choice for the seventh and eighth tasks is made using software toggle buttons. When switching from platform control to elbow angle control, the eighth task variable in the slider window changes from the platform position to the elbow angle, reflecting the variable under user control, 'l'he system also provides the capability of utilizing the arm redundancy in order to avoid collision with workspace obstacles in tasks such as reaching safely through an opening, When the obstacle avoidance task is activated, the obstacle avoidance window shows the distance of the arm from the center of the opening, and the entry angle. The entry angle is defined as the angle between the arm link entering the opening and the normal to the opening. The sliders display the current values, and the maximum allowable values. When the maximum allowed values are exceeded, the obstacle avoidance task is aborted. 'l'he system allows the operator to continue to move the arm, however, the responsibility for obstacle avoidance falls on the operator.

The current implementation of configuration control enables the user to specify the arm "posture" as well as its tool position and orientation. The typical choice for free space motion for the seventh and eighth tasks is arm angle and elbow angle control. 'l'he use of arm angle control allows the user to specify the placement of the elbow, while elbow angle control frees the user from worrying about reaching the limit of the workspace of the arm along the platform axis. This ability for direct and explicit control of the physical configuration of the arm during a specified hand motion provides considerable flexibility in executing tasks that demand high dexterity. This is in contrast to using Jacobian-based methods for redundancy resolution in which the elbow is allowed to move without resti aint *[8].*

'l'he various command and control modes provide considerable flexibility for operation of the mobile dexterous robot. The mode of operation can be changed on-line by the user at any time based on the task at hand. The control system greatly increases the up-time of the arm by being robust to singularities and joint limits. A command line interface is also provided to operate the arm directly from the VME environment in case of the IRIS breakdown.

5 Software Architecture

In this section, we discuss the software components of the VME environment used for real- time control of the manipulator. All of the software executing on the VME environment is written in the C language. Code is developed on a SUN Spare 10 UNIX computer utilizing Wind River System's VxWorks development environment. The development environment consists of a C compiler, a r-emote symbolic debugger, and the Stethoscope real- time monitoring tool. The code is downloaded through Ethernet to the target processor boards foı execution.

Figure 9 shows the software structure of the VME-based controller. The VME chassis hosts two Motorola MVI 67 MC68040 CPU cards that perform all the necessary computations to provide real--time control of the manipulator and the base platform. The user *interface (ui) task* interfaces with the high-level system residing in the IRIS to receive user commands and to send acknowledgment and state information after execution of the commands. The information is routed hi–directionally through Ethernet using the UNIX socket protocol. Once a command is received from the IRIS, the *ui task* parses the command and then writes appropriate command information onto the shared memory card to pass along to the other tasks. All commands from the IRIS are acknowledged by the controller. Every reply from the controller contains the state of the system which includes information such as sensor data, current joint angles, current mode, and Cartesian task values. 'l'he information also includes the current parameters that the system is using such as the elbow angle margin and the hand controller gains. The state of the controller also indicates whether joint limit tasks arc activated, whether the arm power is on, and the current seventh and eighth tasks chosen.

The *hand controller (hc) task is* designated to perform data acquisition. It controls the activities of the Analog-to-l) igital (A/Ii) converter boards which are used to read in joystick inputs and sensor data. 'l'he first A/I) board reads in the voltage outputs of the six potentiometers on the joystick. In addition, it monitors the three switches on the rotational grip joystick. The second A/D board reads in the sensor data from the temperature sensor, gas sensor, and the two proximity sensors.

The *control (ctrl) task* performs real- time trajectory generation and kinematic computations. Both automated and teleoperated moves are supported in joint mode. in Cartesian mode, the arm can be moved with reference to its end-effector frame (tool mode) or an absolute base frame (world mode). *Tool* mode enables the user to move the joints of the robot in a coordinated manner such that the user has the notion of moving the end-effecter as if it is being held by the user's hand (e.g., holding onto a screwdriver and moving the handle to control the tip of the screwdriver). *World* mode is used when the user wishes to move the robot with respect to a fixed user-defined frame. The forward kinematics and Jacobian computations, damped- least- squares computation, and the Cholesky decomposition computation have been timed to take approximately 1.1 *msec* to complete. In this process, the differential desired Cartesian commands (ΔX_d) are converted to differential desired joint commands $(\Delta\theta_d)$, which are then integrated and the desired joint angles θ_d are sent to the arm control unit for execution.

The *robotics research servo (rrs) task* is designated solely to execute the arm interface driver at every servo cycle, thereby maintaining constant communication with the arm control unit. The arm control unit has the Electronic Servo-Level Interface, which allows the user to communicate directly with the joint servo motors through dual-port memory locations on the Multibus. The task communicates with the arm control unit at the maximum possible rate of $400Hz$, i.e. every $2.5msec$. Each joint servo motor can be independently commanded in any of the four modes: position, velocity, torque, and current, This feature enables the operation of the robot under both kinematic and dynamic control schemes, and therefore facilitates validation of a variety of arm control laws. The feedback information such as the actual position, velocity, and torque values are also accessed from the dual-port memory. Presently all seven joints are commanded in position mode. The driver performs all necessary handshakes with the arm control unit software and conversion of data into appropriate formats. In addition, joint position and velocity limits are also checked at each cycle for safety reasons,

6 Experimental Results

In this section, we present experimental results of the robot control system. Seven experiments are conducted. The first experiment, which is labelled the "self-motion" experiment, shows the tracking of arm angle command with time. The effects of W_v and K on the tracking performance are shown in the next three experiments, which are labelled "track]", "track2", and "track3". The avoidance of joint limits is demonstrated in the fifth experiment, called "wrist". The next two experiments, called "xlimit 1" and "xlimit2", show the behavior of the configuration control system near the workspace boundary for two values of W_v. The final experiment labelled "platform" demonstrates how base mobility can be utilized to appropriately place the arm in order to reach a target wrist position. In all experiments, W_t is set to 1 without loss of generality, since the ratio of W_v to W_t is the relevant parameter. Positions all refer to the tool tip, and are expressed in the Cartesian world coordinates. orientations are specified in ZYX fixed angles [2] in the Cartesian world coordinate system. The base is moved only in the "platform" experiment. It is kept stationary at a position of –257 cm for the first seven experiments, i.e. the eighth task is specified to be the platform position.

6.1 Self–Motion Experiment

in the "self- motion" experiment, the initial position of the arm is: $x =$ –340 cm, $y =$ –50 cm, z = 120 cm, $Rotx =$ 90°, $Roty =$ 0°, $Rotz =$ 90°, and ψ = 175°. The arm is commanded to go to a position of $\psi_d = 0°$, i.e. a change in arm angle of – 175°, in 1^0 scc.ends while keeping all the hand coordinates constant. In this experiment, we set W_v = 0.005, and K = 1.0. Figure 10a shows a plot of arm angle vs time and a plot of the 1)1,S error with time is shown in Figure 10b. The units for the DLS error are meters for x, y, and z; and radians for the hand orientation and the arm angle. Good tracking is exhibited, with a maximum transient error of approximately –0.03 radians or 1.7° for the arm angle.

6.2 Track1, Track2, and Track3 Experiments

In the "track1", "track2", and "track3" experiments, the initial position of the arm is: x = —300 cm, y = —50 cm, z = 130 cm, $Rotx =$ 9 0 0' $Roty = 0°$, $Rotz =$ 9 0 0' and ψ = 135°. The arm is commanded to go to a final position of x = -370 cm, $z = 60$ cm and $\psi =$ 900 in 5 seconds while keeping the other Cartesian coordinates constant. This corresponds to a diagonal move of approximately 100 cm, and an arm angle change of –45°. For "track]", $W_v =$ ().005 and K = 1.0. Track1 is the *nominal* case with suitable values for W_v and K. Track2 and Track3 study the effects of W_v and K on the tracking performance, respectively. The effect of W_v is demonstrated in "track2" by using a W_v of 0.05, and a K of 1.0. "Track3" shows the effect of K by setting W_v to 0.005 and K to 0.1. For each experiment, five plots are shown: z vs time, x vs time, ψ vs time, x vs z, and DLS error versus time. These are shown in Figures 1 la-, 1 1e for "track]", Figures 12a-- 12e for "track2", and Figures 13a- -13e for "track3".

The "track]" experiment shows how the arm coordinates track the commands, with a small but nonzero value for W_v. *These* empirically derived values of W_v and K give good tracking while providing protection against high joint velocities near singularities. Note that tracking appears to be quite good except for a fixed offset from the desired trajectory. This fixed offset is especially apparent in Figures 11 b (z vs t), 11c (ψ vs t) and 11d (z vs x). This offset can be attributed to the fact that the Cartesian coordinates X used in the feedback loop of the control diagram (Figure 3) arc obtained using the computed joint setpoints θ_d as opposed to the measured joint angles θ_m from the RRC servos, The joint setpoints θ_d are used because prior to running at a sample rate of 400 Hz, the control loop was unstable when the measured joint angles θ_m were used due to the additional phase lag introduced by the arm-plus-servo dynamics. Figure 1 1e shows a maximum transient error of approximately 2 cm in x.

In the "track2" experiment, W_v is increased from 0.005 to 0.05. The experimental results are shown in Figures 12a- 12e. As can be seen from the plots, increasing W_v has the effect of incurring much larger tracking errors. The maximum tracking error in x is now 19 cm. The increase in W_v has the effect of limiting joint velocity even further at the expense of losing tracking accuracy. Note that the arm does eventually reach its steady-state position after approximately 10 seconds. Thus W_v produces a transient effect, only taking effect when joint velocities are high. In the steady -- state, low joint velocities are required when the arm is near its final commanded configuration, and the influence of W_v is greatly diminished.

In the "track3" experiment, W_v is set back to 0.005 and K is decreased from 1.0 to 0.1, Thus the effect of K can be seen by comparing the "track3" plots (Figures 13a-13e) with the "track]" plots (Figures 1 la- 1 1e). The "track3" plots show generally good tracking, similar to "track]", the difference is in the speed of convergence to the final value in the steady–state. Convergence is much slower in the "track3" experiment. This shows that K has a direct effect on the speed of convergence to the commanded values.

6.3 Joint Limit Avoidance Experiment

'] 'he "wrist" experiment demonstrates the configuration control scheme in action when a joint limit (the wrist joint 6, in this case) is reached. Joint 6 on the RRC arm has a physical range of – 180° (hand folded onto itself) to 0° (hand straightened out). The weighting for the joint limit avoidance task W_ζ is given by equation (1 7). W_ζ is a smooth function that starts at O when the joint is within 10° of its limit (the "soft limit"), and increases to a maximum of W_{max} = 10 when the joint is within 3.5° of its limit (the "hard limit"). Thus the joint limit avoidance task is activated when joint 6 is less than –lO°. The initial position of the arm is: x = --330 cm, y = –50 cm, z = 120 cm, $Rotx$ = 90°, $Roty$ = 0°, $Rotz$ = 90", and ψ = 135°. The arm is commanded to go to a $Rotx$ position of 30° in 12 seconds, with W_v = 0.005 and K = 1.0. In this configuration, the $Rotx$ angle corresponds to the pitch of the tool. The trajectory makes the arm pitch up, and the wrist hits its joint limit during the motion. Figure 14a (joint 6 vs time) shows how the wrist joint reaches the "soft" joint limit of –10°, then eventually backs away from –10". What happens physically is that the upper–arm roll joint (joint 5) makes a 180° change, which flips the wrist joint so that a decrease of the wrist joint now corresponds to pitching up. Note that none of these joint motions are explicity commanded by the user, the motions '(fell out" of the configuration control scheme. Figure 14b ($Rotx$ vs time) shows a loss of tracking due to joint 6 hitting its joint limit, but the arm eventually recovers and reconfigures itself to reach the desired final position.

6.4 Xlimit1 and Xlimit2 Experiments

The next two experiments, "xlimit1" and '(xlimit2°, show the behavior of the arm at the workspace boundary. From an initial starting position of: x = –330 cm, y = –50 cm, z = 120 cm, $Rotx$ = 90°, $Roty$ = 0°, $Rotz$ = 90°, and ψ = 135°, the arm is commanded to move to x = –380 cm in 10 seconds, keeping all other task variables constant. in "xlimit1", W_v *is* set to 0.01, and K = 1.0. W_v *is* increased to 0.05 in "xlimit2", while K remains at 1. The final Cartesian position represents a target that is not reachable by the arm. In “xlimit1”, the arm shows good tracking (Figure 15b) till close to the final position. Figure 15c (DLS error vs time), however, shows that the DLS error is not converging to zero since the final position is unreachable. The plot does show that since the weighting W_t is equal for all tasks, the DLS errors converge to approximately the same order of magnitude, i.e. the tracking- error is distributed among the various tasks equally. In the “xlimit2” experiment, we observe larger maximum errors due to the larger value of W_v. We also observe larger steady- state errors. When the arm is commanded to go to an unreachable target there are two conflicting tasks. ‘l’he KE term tries to move the arm to the final target, while the W_v term retards joint velocities. Thus in “xlimit2” the larger W_v term is able to retard the effect of K in the steady-state more than in the “xlimit 1” experiment. ‘l’his is also reflected in the plot of the elbow angle (joint *4)*. In “xlimit 1“, the elbow angle is closer to its fully stretched out configuration compared to the “xlimit2” experiment.

6.5 Platform Experiment

The final experiment demonstrates how Last mobility can be utilized to appropriately place the arm in order to reach an otherwise unattainable target wrist position. In this experiment, the user specifies the task weighings W_t of 1 and feedback gains K of 1 for the wrist and arm angle control tasks, joint velocity weightings W_v of 0.005, and elbow angle margin δ = 30°. Starting from the initial wrist position of x = –307.5 cm, y = --62.4 cm, and z = 120.0 cm, and the initial arm angle ψ = 45°, the wrist is commanded to move to the final position of x = —392.5 cm, y = —*62.4* cm, and *2 = 60.0* cm in *35* seconds while ψ is kept constant. Note that the target wrist position is beyond the reach of the arm if base mobility is *not* activated. ‘l’he elbow angle control is selected as the eighth task, Figure 17 shows the experimental results for the system. The plot shows that when the elbow angle exceeds 120°, the platform starts to move automatically and brings the elbow angle back to approximately 90°, Notice that since the wrist velocity is greater than the base velocity, the elbow angle exceeds the user-specified range *60°* $\leq\phi\leq$ 1200 momentarily until the base movement has sufficient time to compensate for the wrist motion. I’bus, the base mobility of the arm is used effectively to prevent the arm from reaching its workspace boundary.

7 Conclusions

A real--time control system for a mobile dexterous seven DOF manipulator is described in this paper. A kinematic analysis of the platform shows how the base mobility can be used to maximize the manipulability of the end- effector. Experimental results are presented showing the behavior of the end-effector while executing free- space motion. Further experimental results demonstrate how the additional tasks in the configuration control scheme are used to execute a self-motion trajectory, perform joint limit avoidance, and singularity avoidance at the workspace boundary. Finally, experimental results are also presented to demonstrate how the base mobility can be utilized for elbow angle control, and thus ensure reachability of the target position,

The manipulator used in this paper is well- suited for tasks that demand positioning and pointing a payload dexterously, such as in the supervised telerobotic inspection project at JPL. The control system provides dexterous motion by controlling the end- point location and the manipulator posture simultaneously. This enables operation of the manipulator in the presence of workspace obstacles and provides the capability to reach safely inside constricted openings. This yields a general- purpose highly-flexible robot control system which is capable of performing many tasks requiring teleoperation or autonomous manipulation in unstructured dynamic environments in both space and terrestrial applications. In fact, although the arm control system has been designed for the telerobotic inspection project, it possesses generic capabilities that can be used for many applications utilizing different hardware.

8 Acknowledgment

The research described in this paper was performed at the Jet Propulsion Laboratory, California institute of '1'ethnology, under contract with the National Aeronautics and Space Administration.

References

[1] M. Brady et al. *Robot Motion: Planning and Control. MIT* Press, Cambridge, MA, 1982.

[2] J, Craig. *Introduction to Robotics:Mechanics and Control.* Addison- Wesley, Reading, Massachusetts, 1986.

[3] K. Glass, R. Colbaugh, I). Lim, and H. Seraji. On-Line Collision Avoidance for Redundant Manipulators, *Proc. IEEE Intern. Conf. on Robotics and Automation, Atlanta, Georgia,* pages 36-43, May 2-61993.

[4] K. Glass, R. Colbaugh, D. Lim, and H. Seraji. Real-Time Collision Avoidance for Redundant Manipulators. *IEEE Transactions on Robotics and Automation,* 11(3):448-457, 1995.

[5] S. Hayati, J. Balaram, H. Seraji, W.S. Kim, K. Tso, and V. Prasad. Remote Surface Inspection System. *Proc. IEEE Intern. Conf. on Robotics and Automation, Atlanta, Georgia,* pages 875-882, May 2-61993.

[6] K. Kreutz-Delgado, M. Long, and H. Seraji. Kinematic Analysis of 7-DOF Manipulators. *Intonational Journal of Robotics Research,* pages 469-481, 1992.

[7] *Y.* Nakamura and H, Hanafusa. Inverse Kinematic Solutions with Singularity Robustness for Robot Manipulator Control. A *SME Journal of Dynamic Systems, Measurement, and Control*, 108, 1986.

[8] R. Schnurr, M. O'Brien, and S. Cofer. The Goddard Space Flight Center Robotics '1'ethnology Testbed. In *Proc. Second NA SA Conference on Space Telerobotics*, pages 491-500, Pasadena, January 1989.

[9] H. Seraji. Configuration Control of Redundant Manipulators: Theory and Implement at ion. *IEEE Trans. on Robotics and Automation*, 5(4):472- *490, 1989.*

[10] H. Seraji. An On- Line Approach to Coordinated Mobility and Manipulation. *Proc. IEEE Intern. Conf.* on *Robotics and Automation, Atlanta, Georgia,* pages *28-33,* May *2-61993.*

[11] 11. Seraji and R. Colbaugh. Improved Configuration Control for Redundant Robots. *Journal of Robotic Systems,* 7(6):897-928, *1990.*

[12] R. Volpe. Task Space Velocity Blending for Real-Time Trajectory Generation. *Proc. IEEE Intern. Conf. on Robotics and Automation, Atlanta, Georgia,* pages *680-687,* May *2-61993.*

[13] C. Wampler and I,. Leifer. Applications of Damped- Least-Squares Methods to Resolved-Rate and Resolved- Acceleration Control of Manipulators. *ASME Journal of Dynamic Systems, Measurement, and Control,* 110:31-38, *1988.*

[14] T. Yoshikawa. Manipulability and Redundancy Control of Robotics Mechanisms. *Proc. IEEE Intern. Conf. on Robotics and Automation,* pages *1004-1009,* March 1985.

Figure 1: MCS Hardware Structure

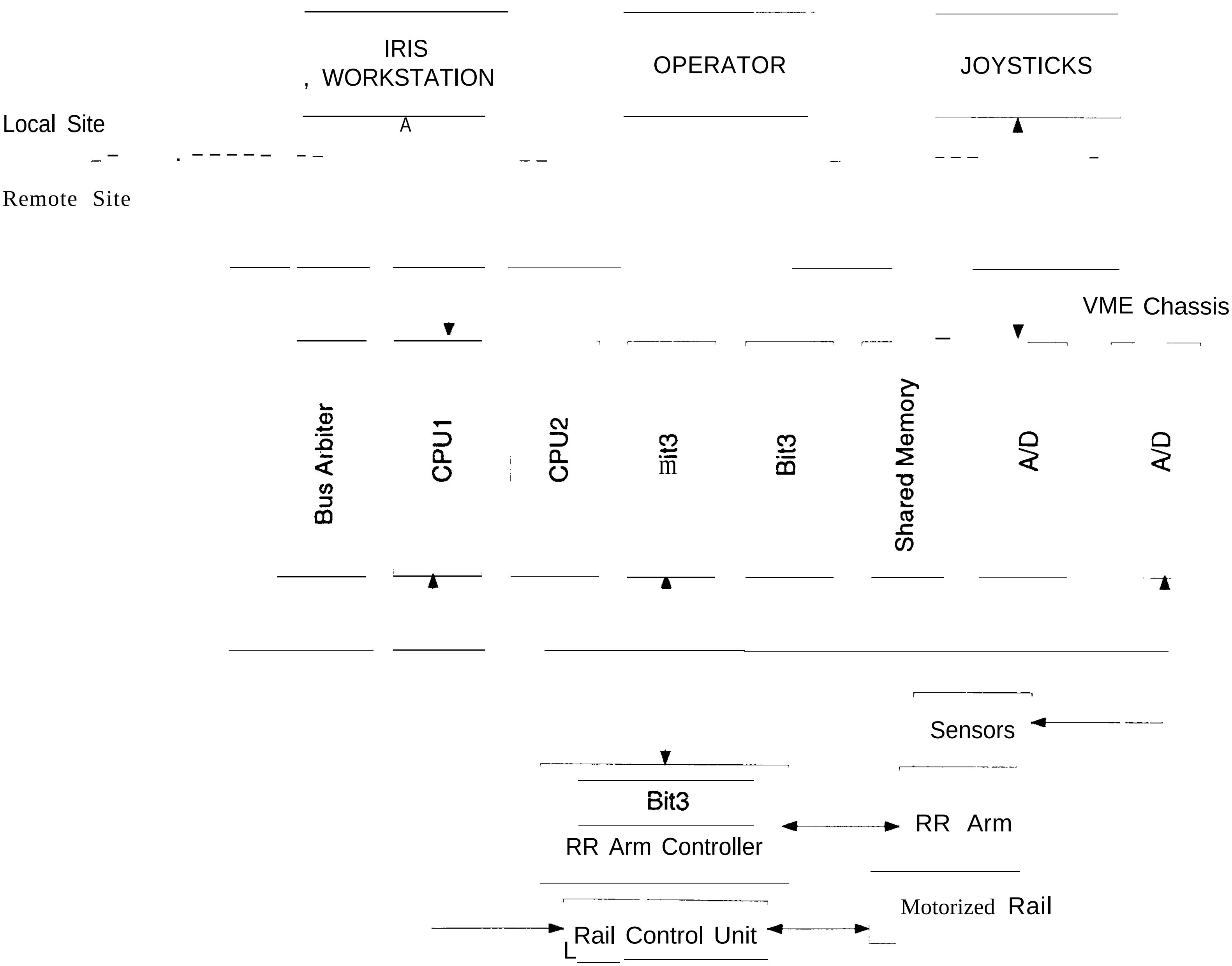
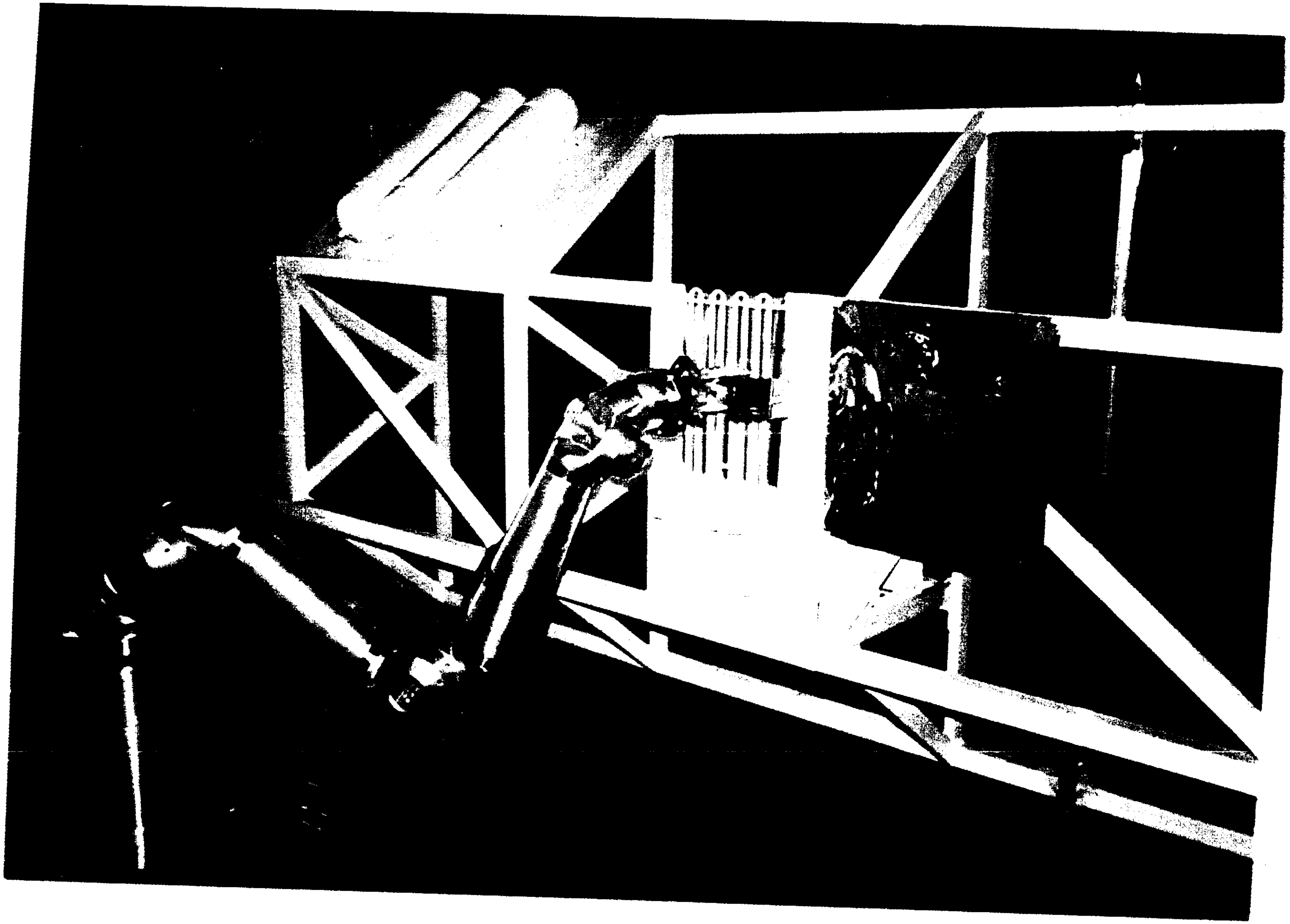


Figure 2: Laboratory Setup



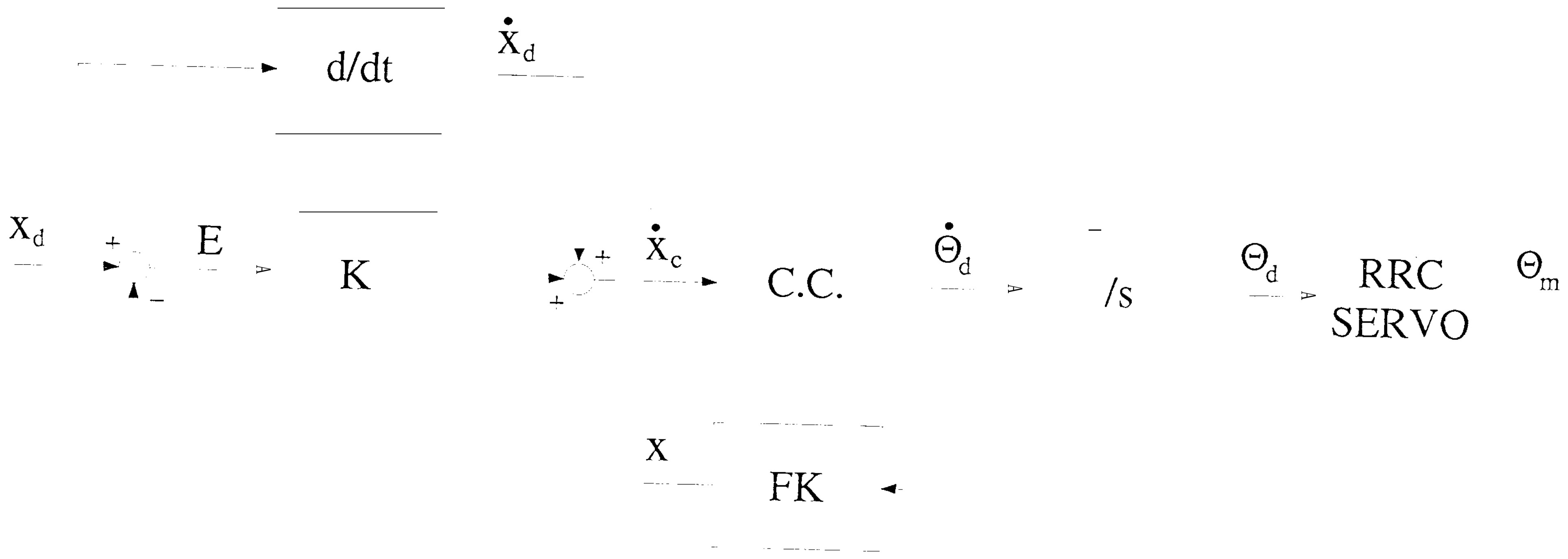


Figure 3. Control Diagram

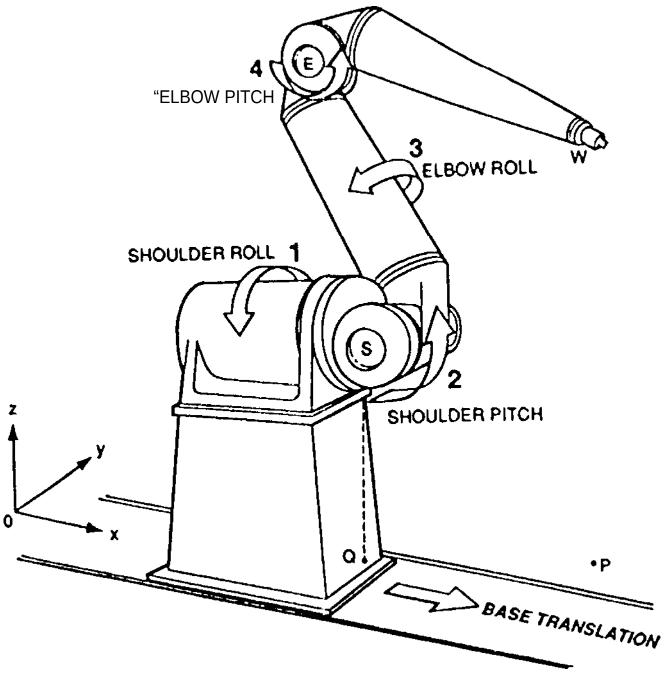


Figure 4: A Robotics Research Arm mounted on a mobile platform

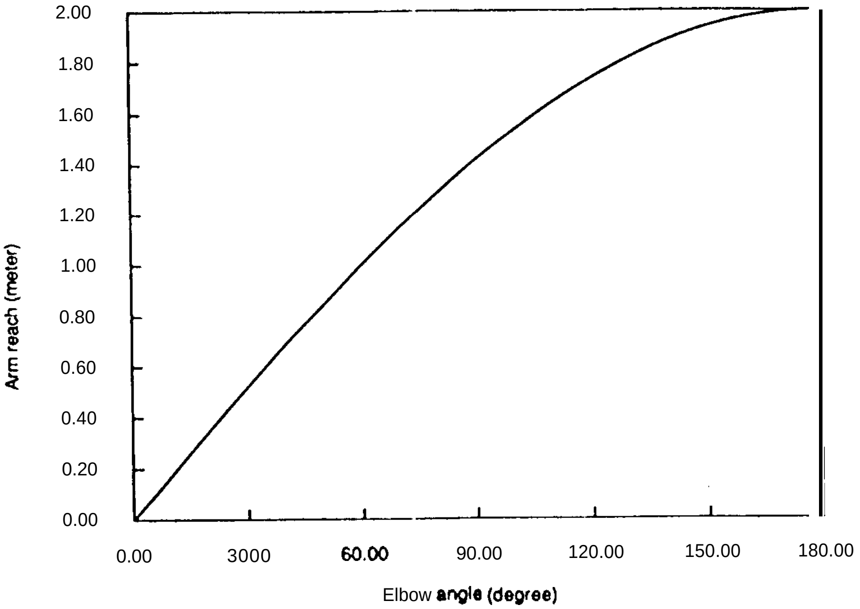


Figure 5: Variation of the arm reach as a function of the elbow angle

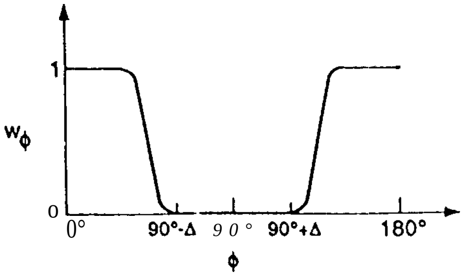
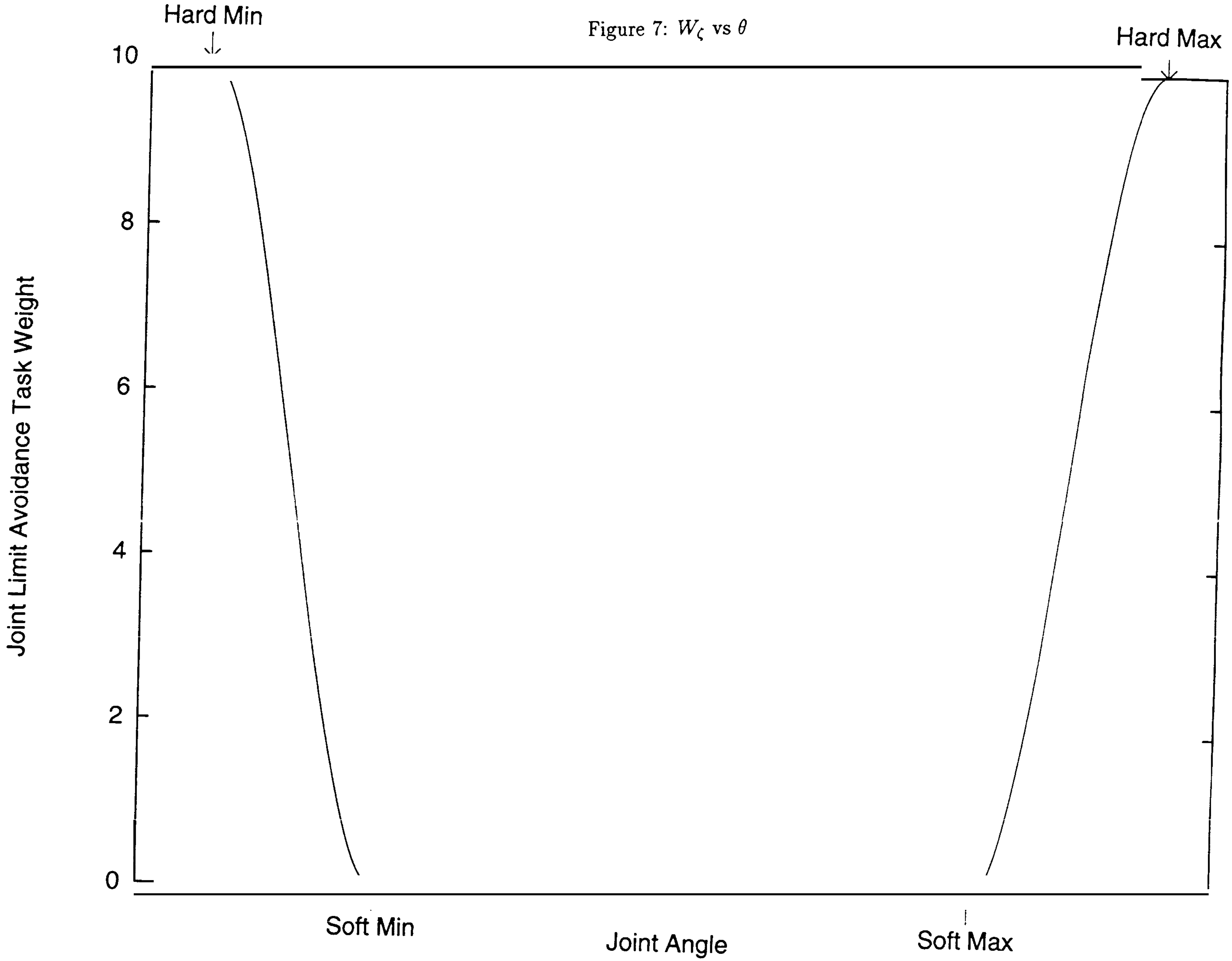
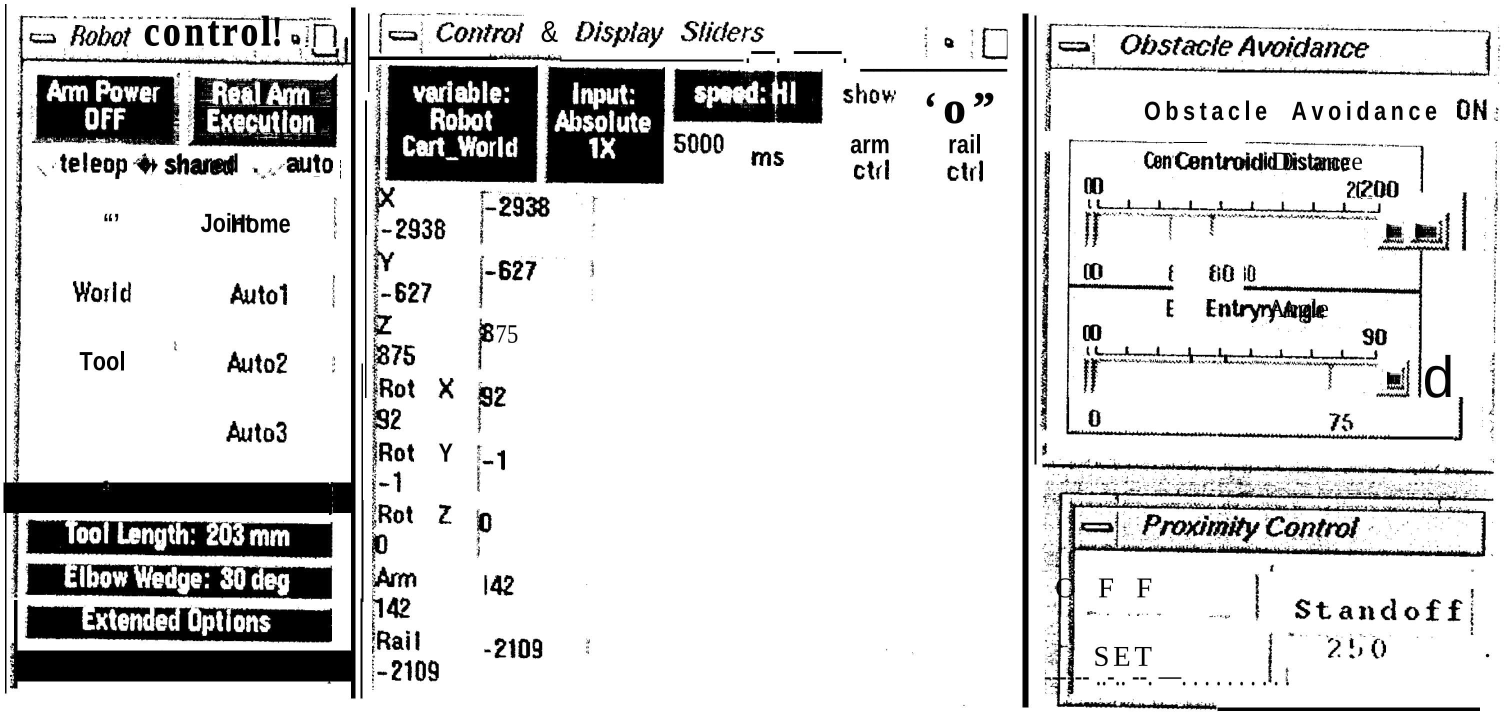


Figure 6: Variation of the elbow task weighting factor as a function of the elbow angle

Figure 7: W_ζ vs θ





Arm Power OFF

Tool Length: 203 mm

Elbow Wedge: 30 deg

variable: Robot Cart_World

Input: Absolute 1X

speed: HI

5000 ms

X -2938 -2938

Y -627 -627

Z 875 875

Rot X 92 92

Rot Y -1 -1

Rot Z 0 0

Arm 142 142

Rail -2109 -2109

Obstacle Avoidance ON

Centroid Distance 80

Entry Angle 75

Standoff 250

Figure 8: Graphical User Interface

Figure 9: Software Structure

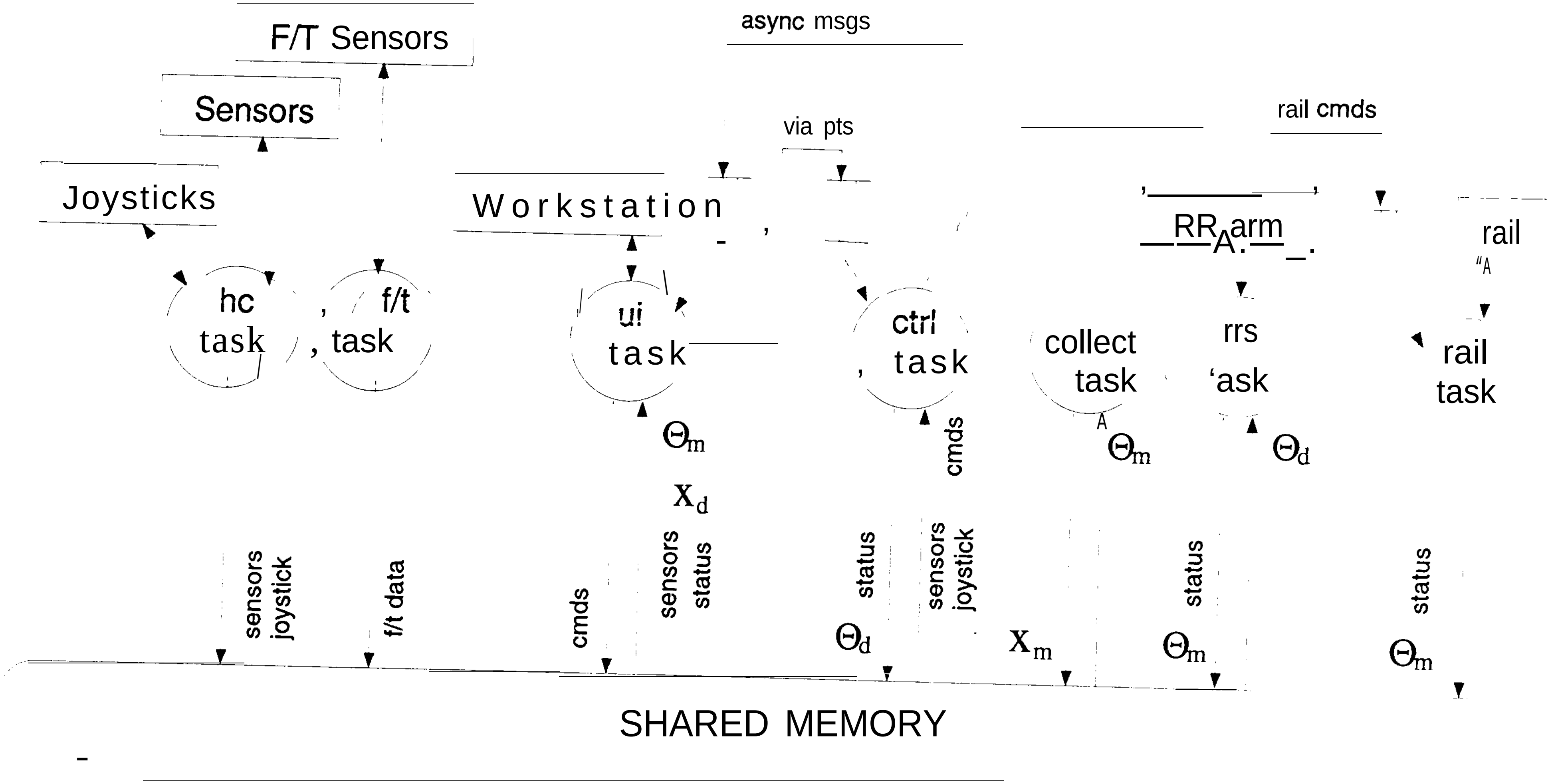


Figure 10a : Self- Motion Experiment ψ vs t

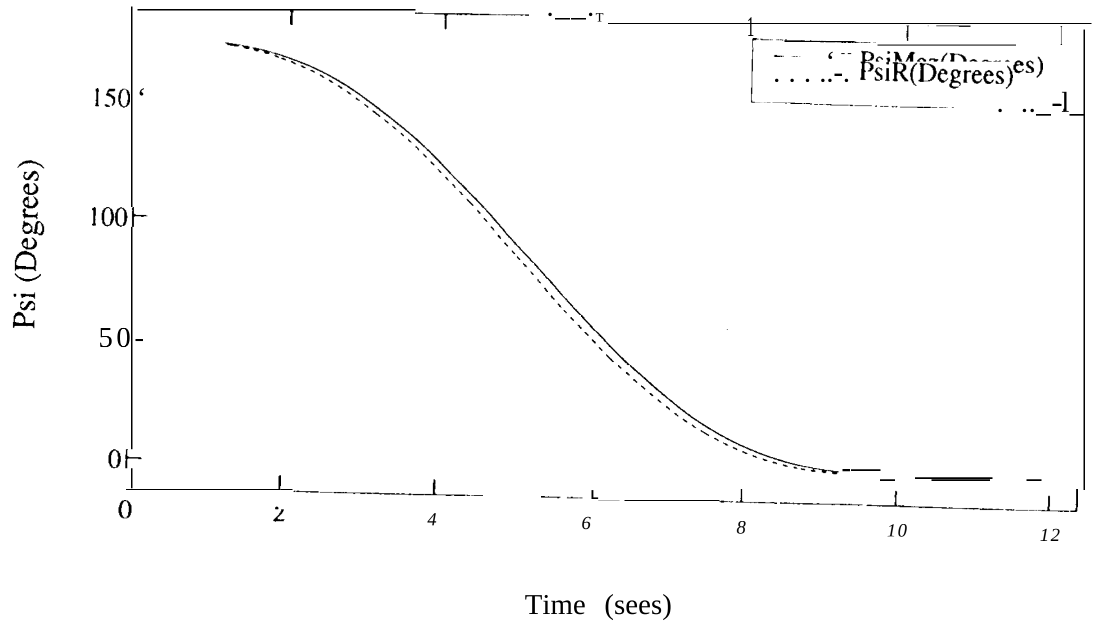


Figure 10b : Self-Motion Experiment DLS error vs t

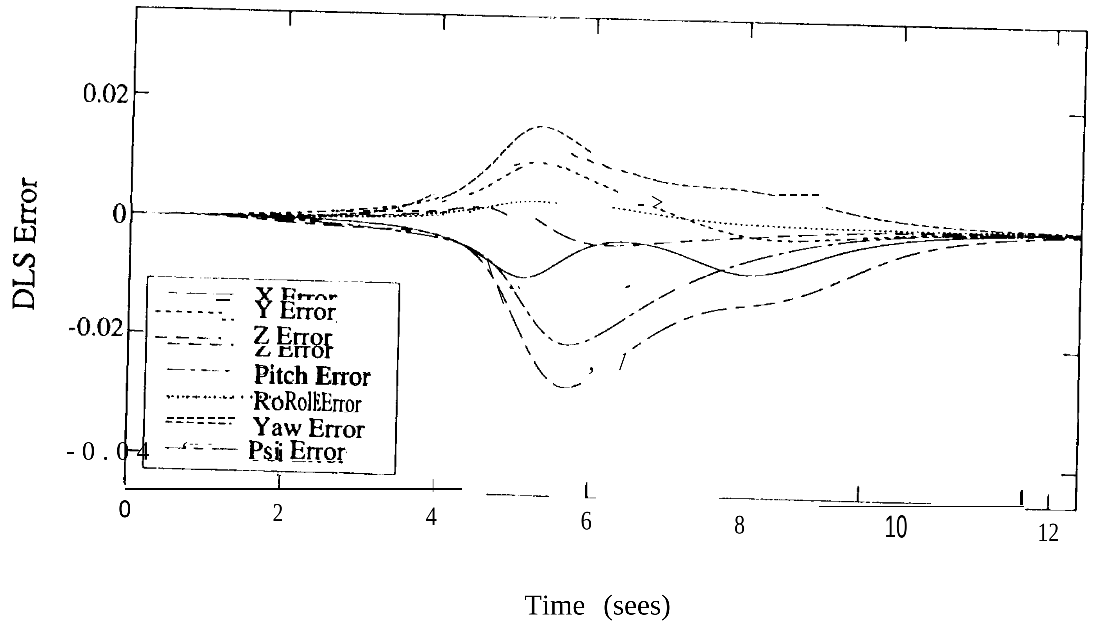


Figure 11a : Track1 Experiment x vs t

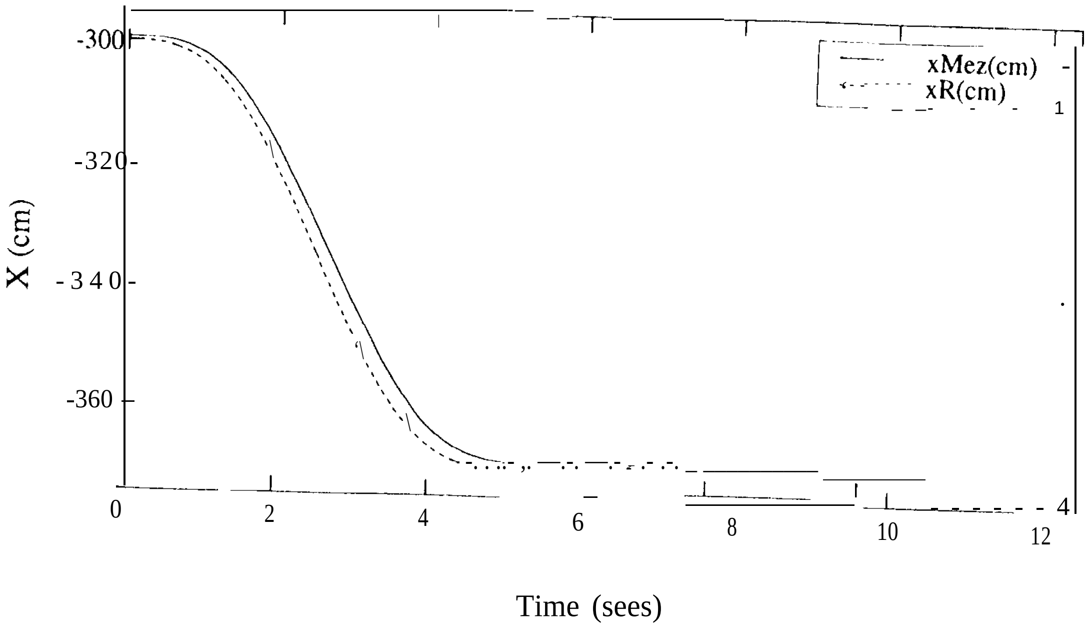


Figure 11 b : Track 1 Experiment z vs t

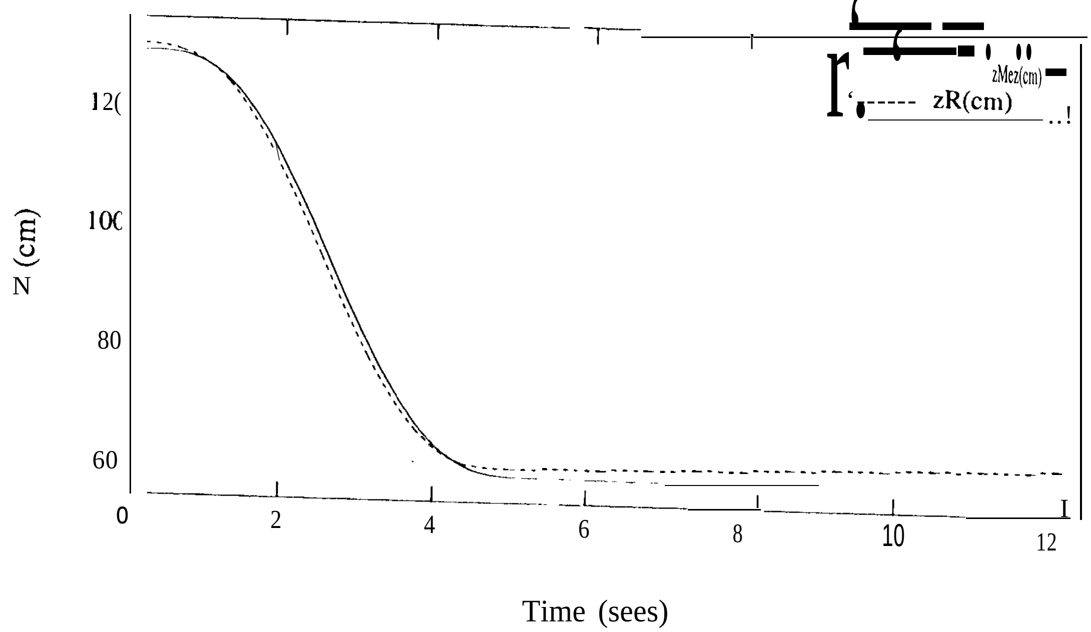


Figure 11c : Track1 Experiment ψ Vs t

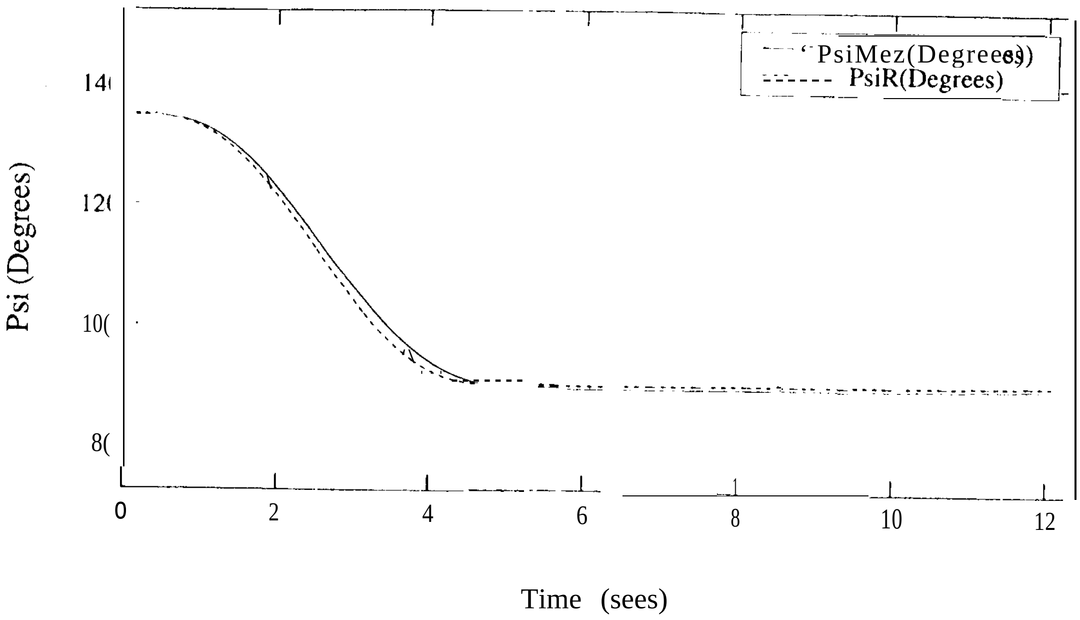


Figure 1 1d : Track1 Experiment z vs x

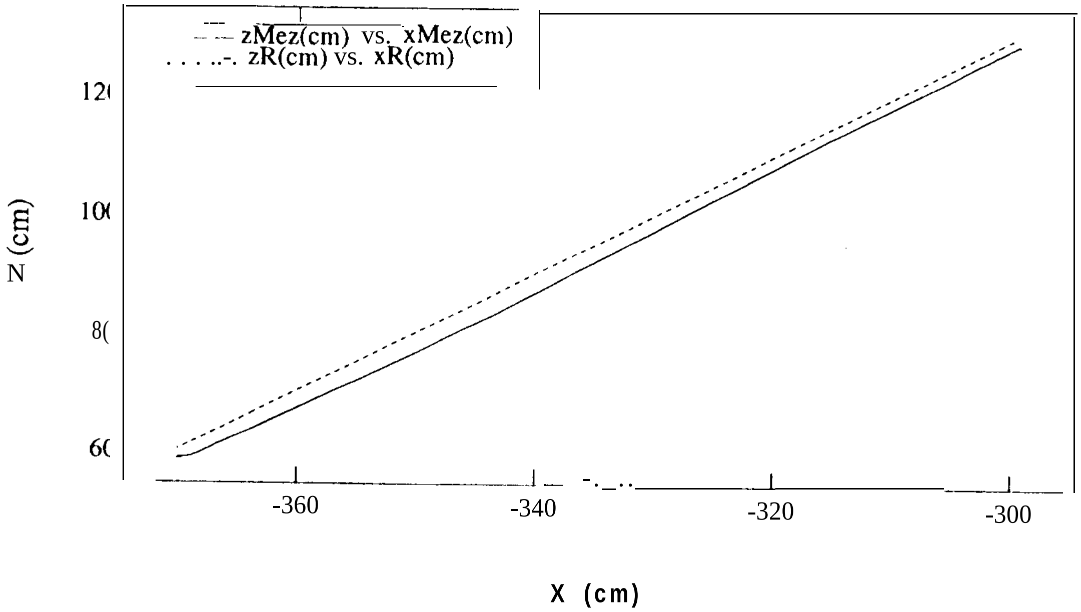


Figure 1e : Track] Experiment DLS error vs t

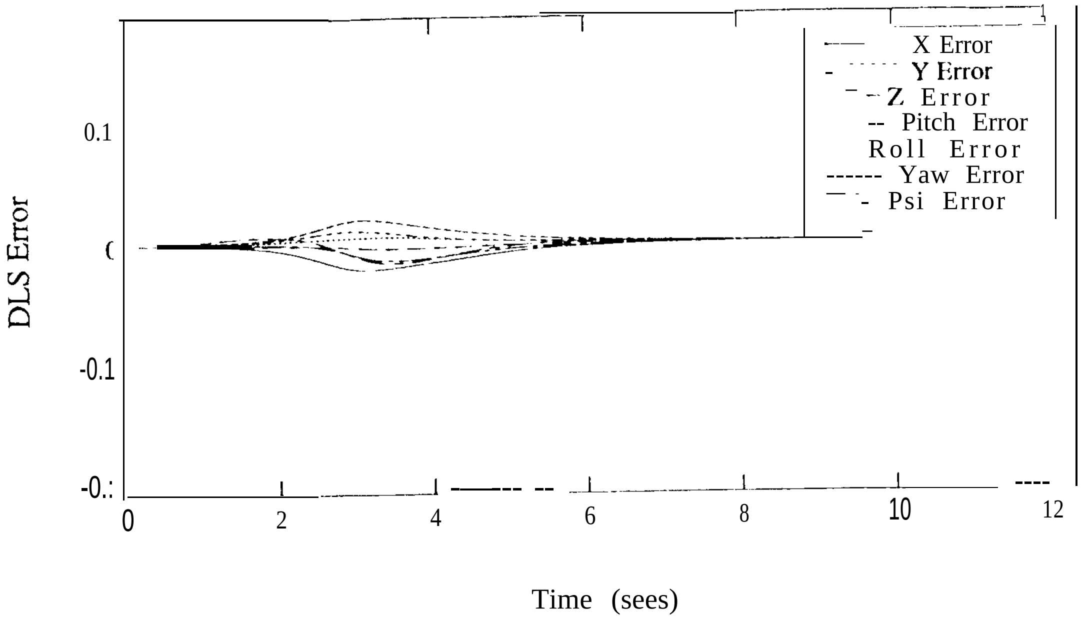


Figure 12a : Track2 Experiment x vs t

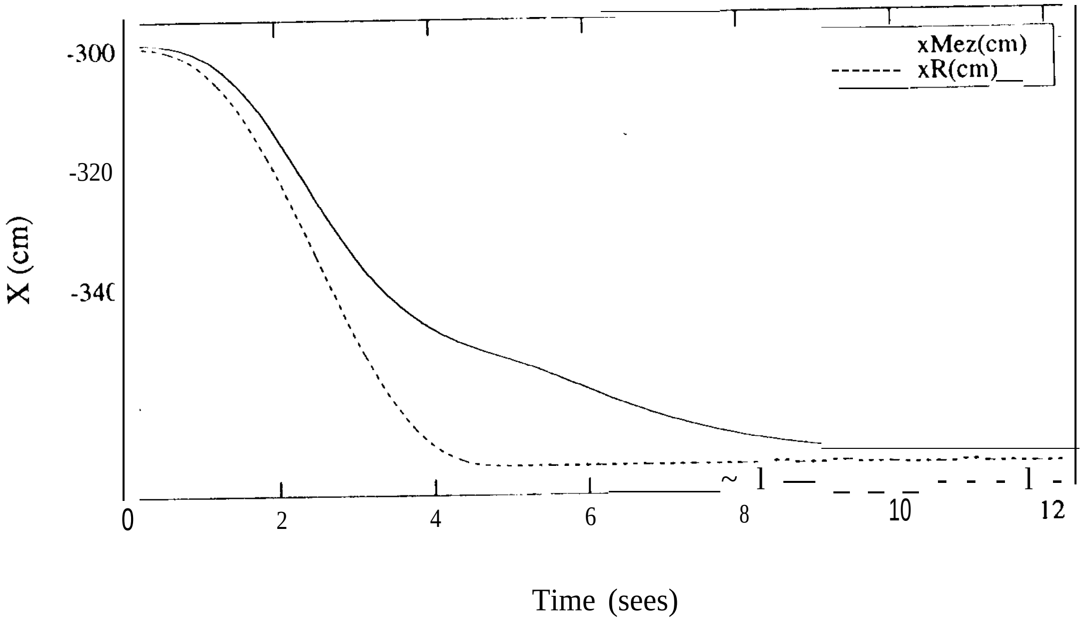


Figure 12b : f rack2 Experiment z vs t

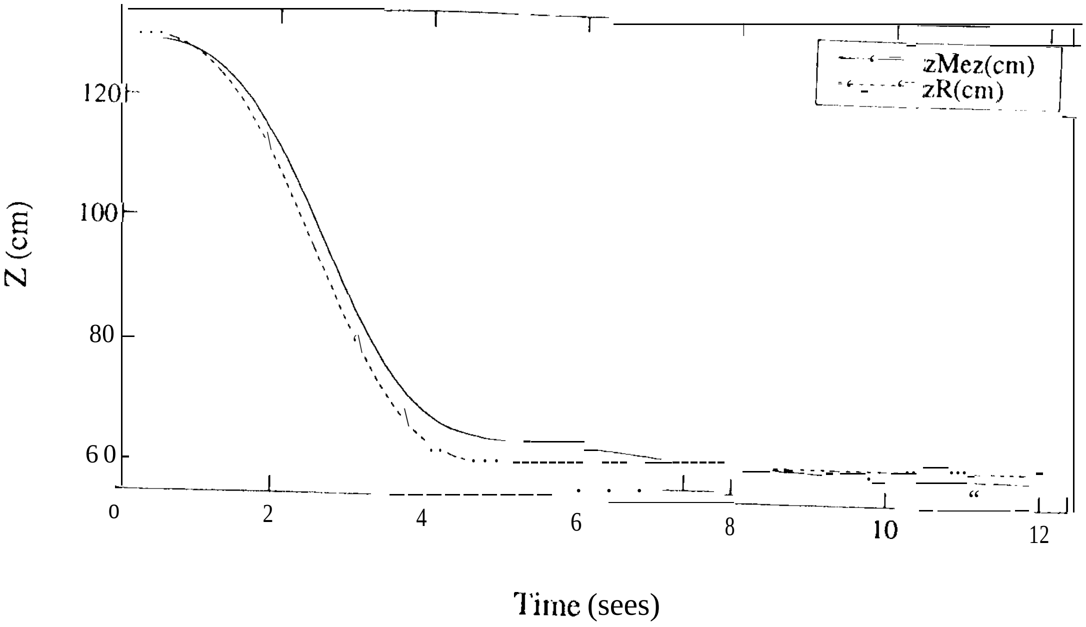


Figure 12c : Track2 Experiment ψ vs t

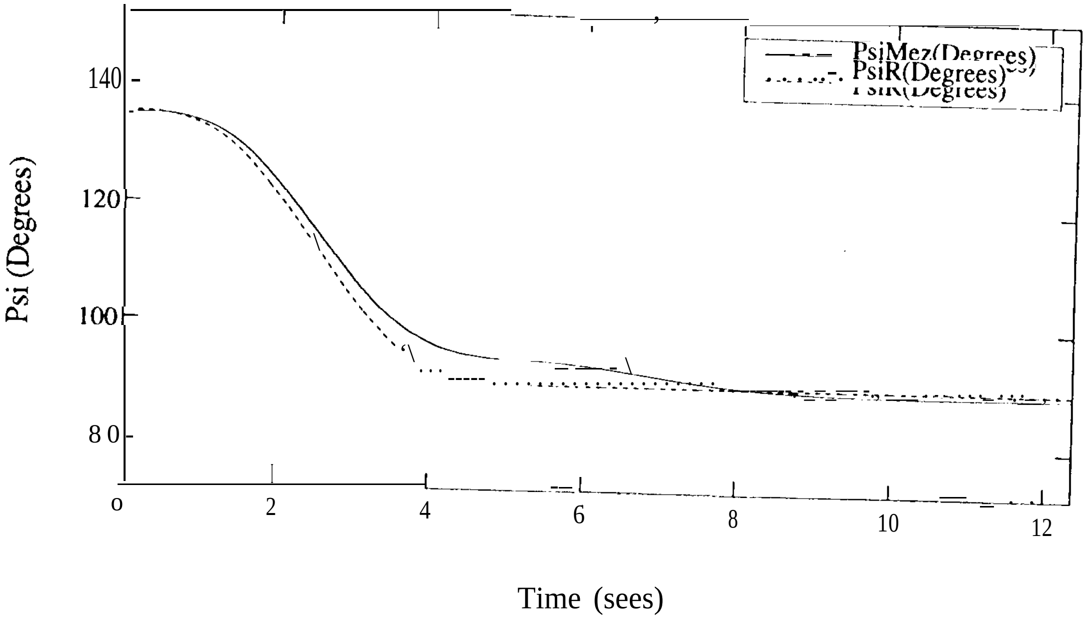


Figure 12d : Track2 l'experiment z vs x

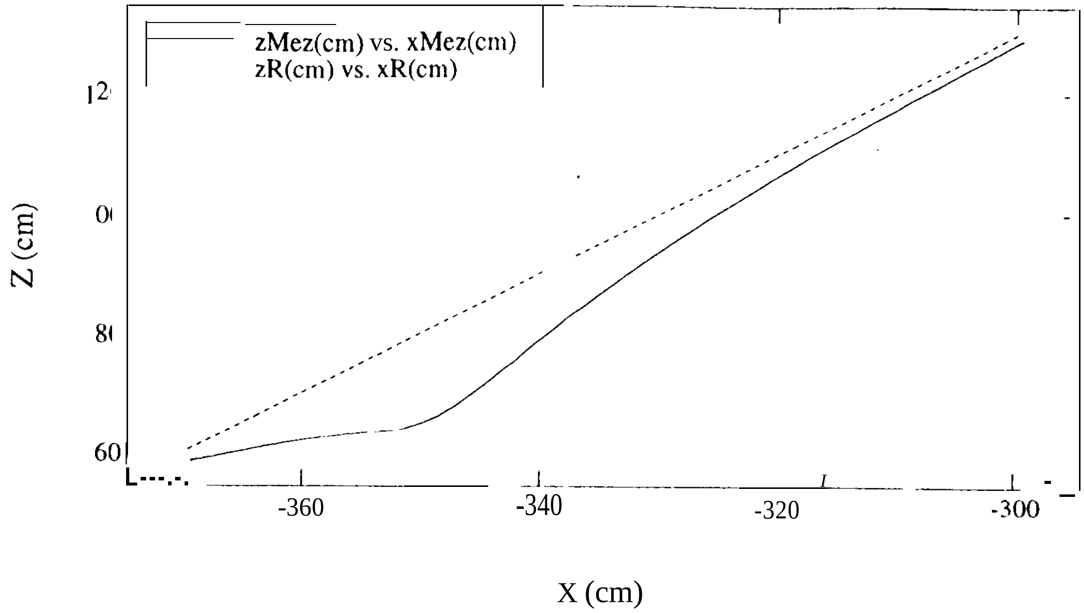


Figure 12e : Track2 Experiment DLS Error vs t

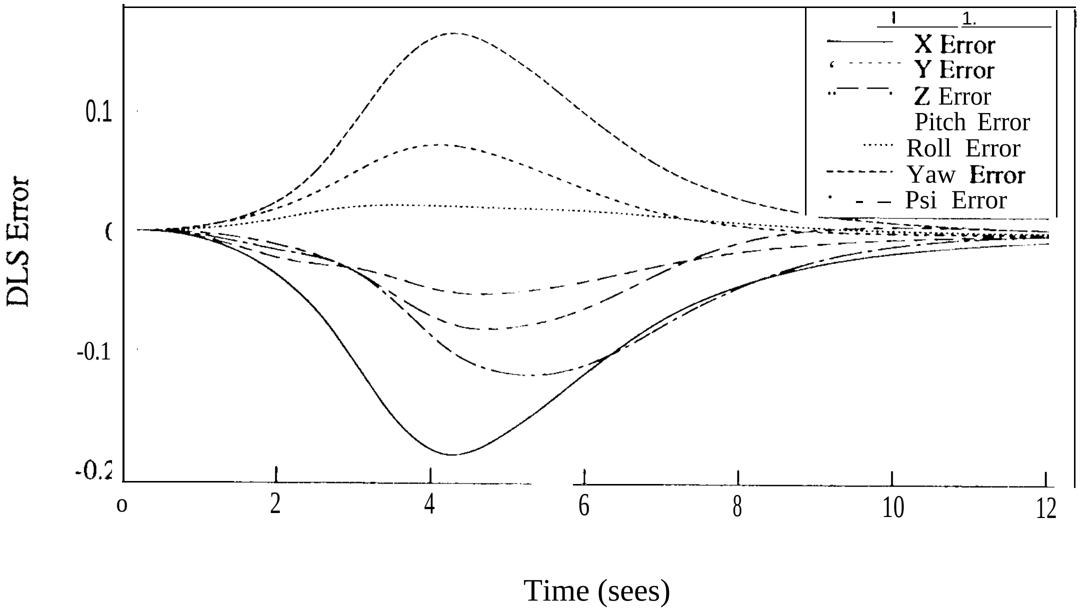


Figure 13a : Track3 Experiment x vs t

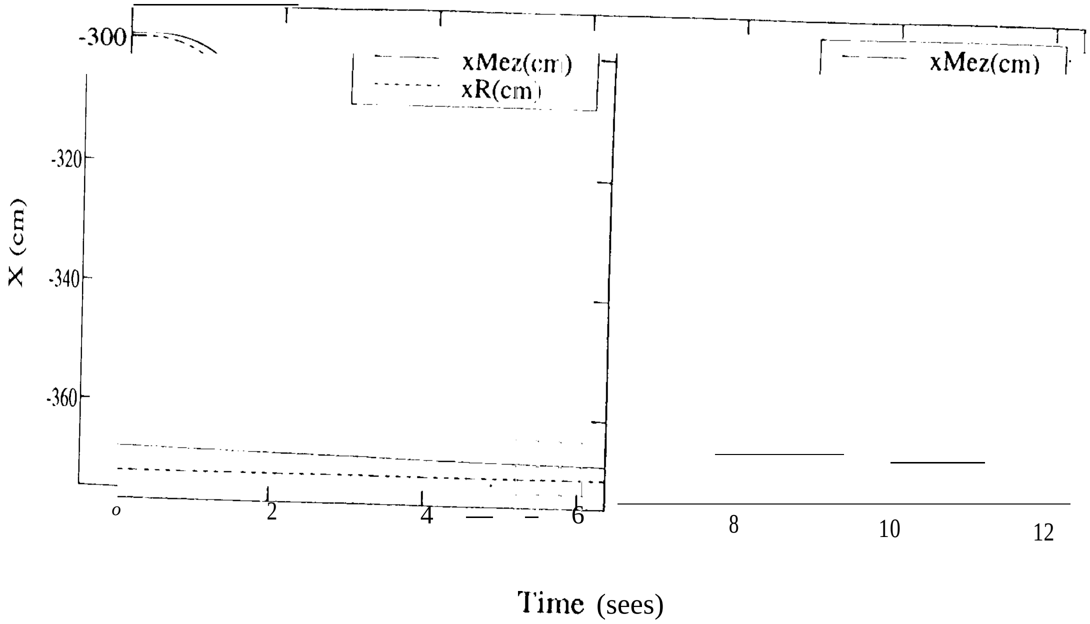


Figure 13b : Track3 Experiment 2 vs t

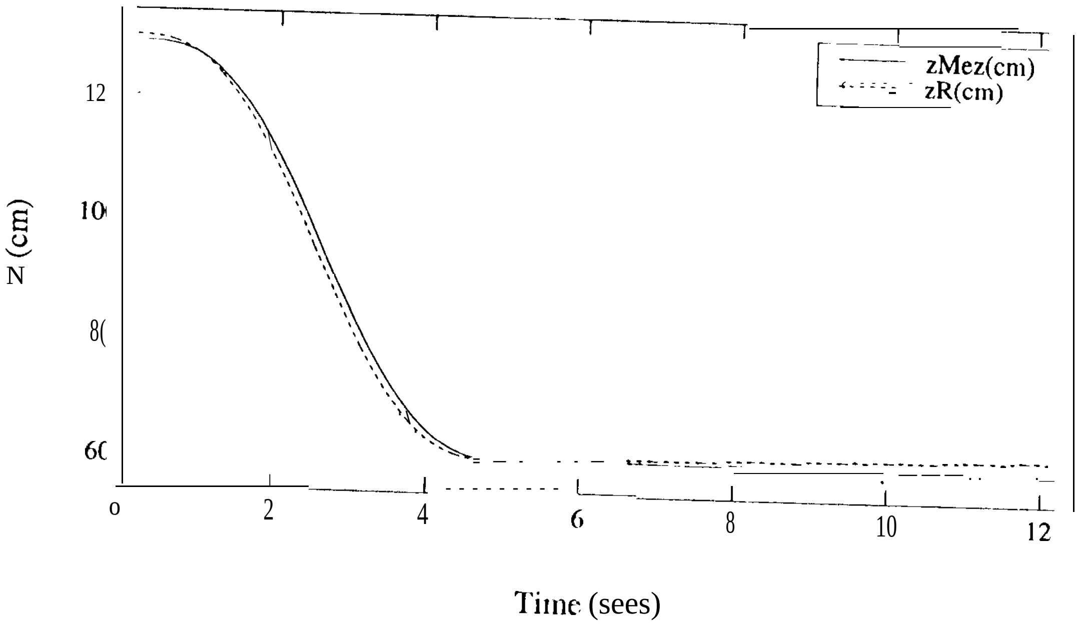


Figure 13c : Track3 Experiment ψ Vs t

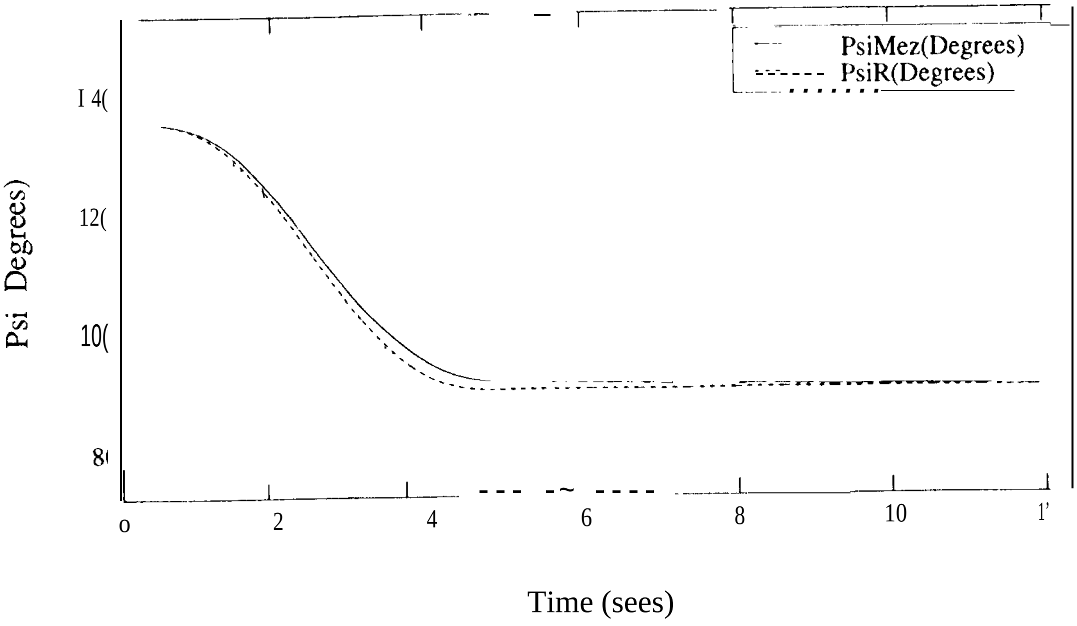


Figure 13d : Track3 Experiment z vs x

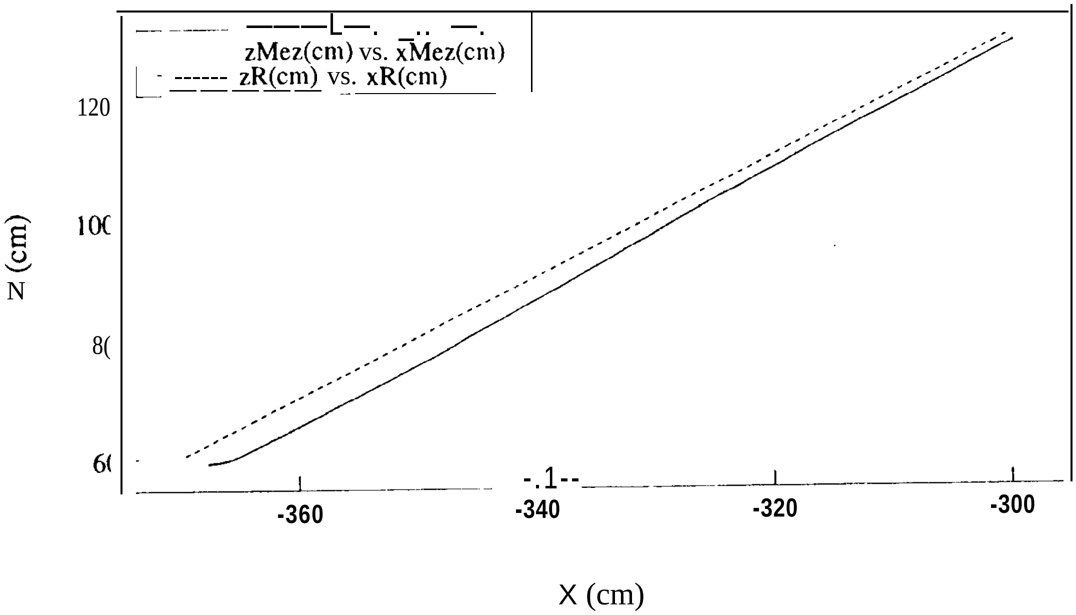


Figure 13e : Track3 Experiment DLS Error vs t

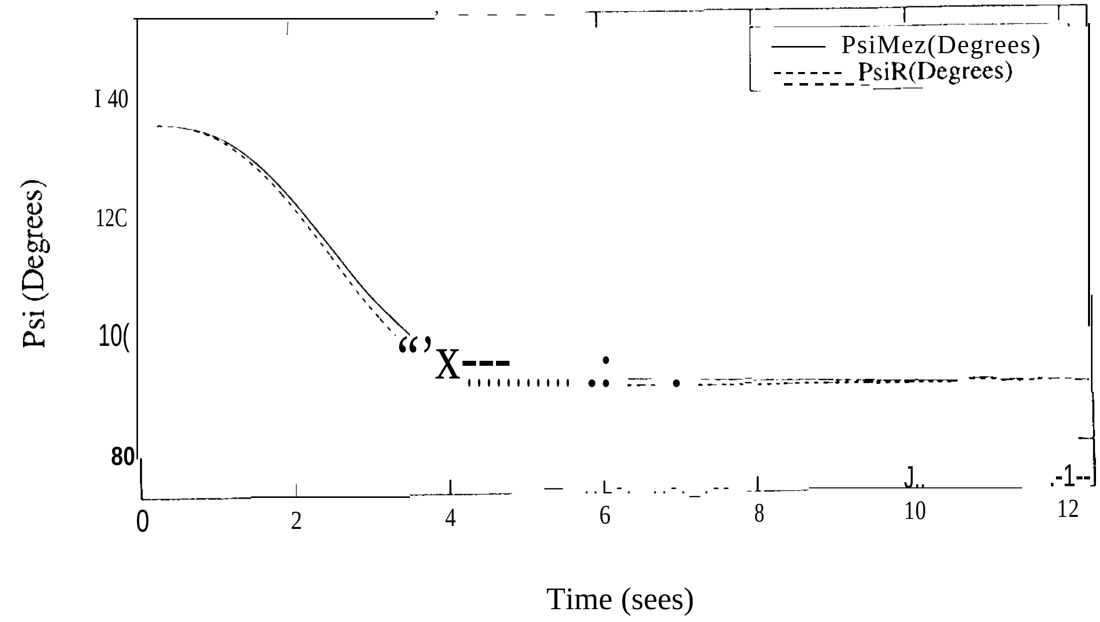


Figure 14a : Wrist Experiment θ_6 vs t

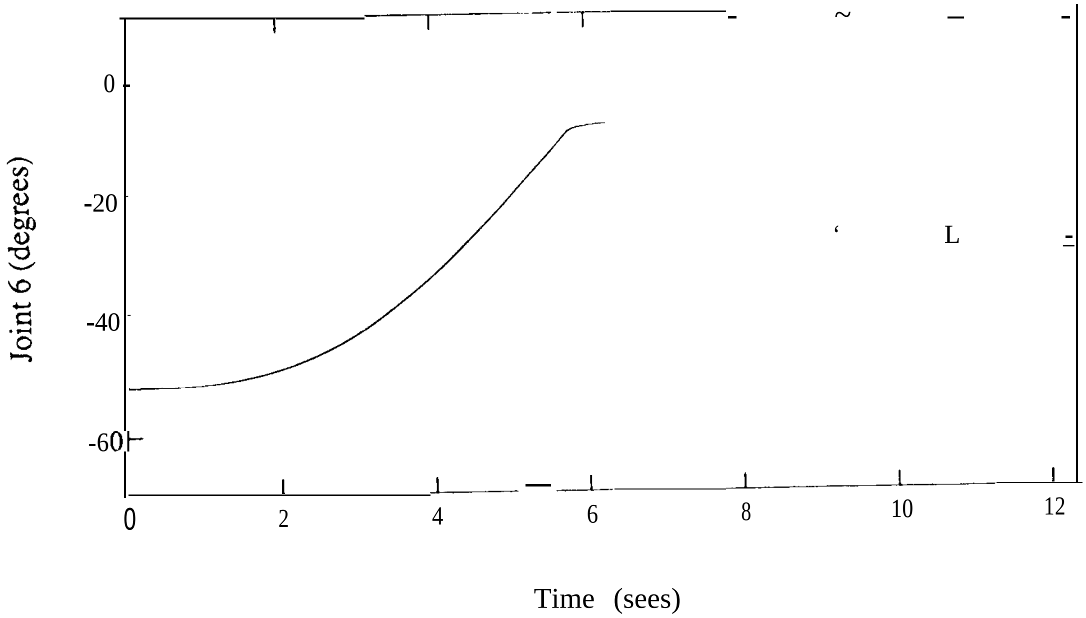


Figure 14b : Wrist Experiment *RotX* vs *t*

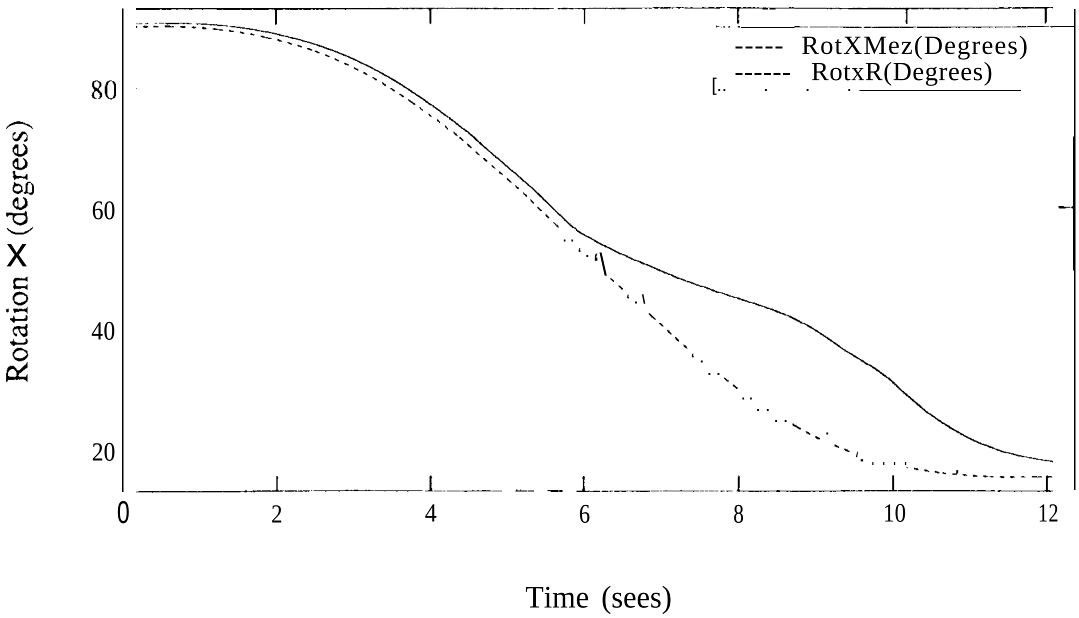


Figure 14c : Wrist Experiment DLS error vs *t*

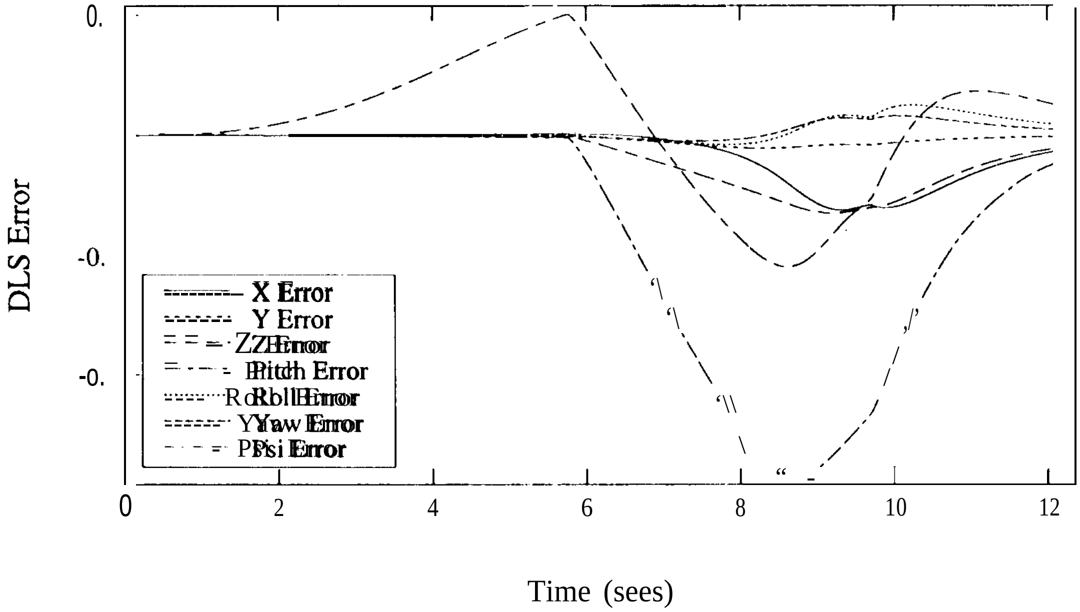


Figure 15a : Xlimit1 Experiment θ_4 vs t

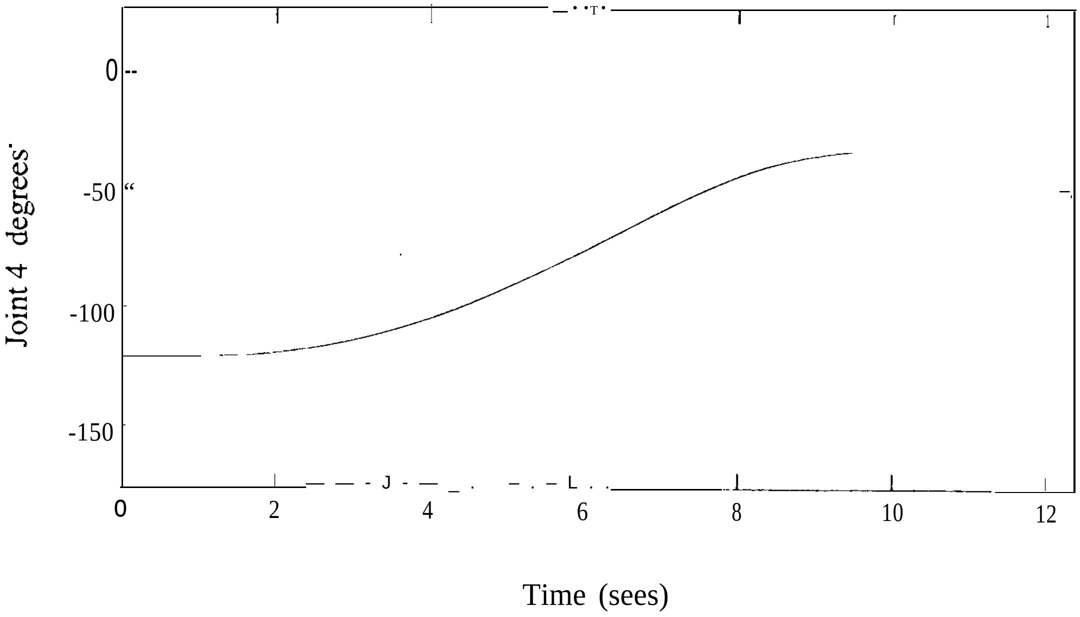


Figure 15b : Xlimit1 Experiment x vs t

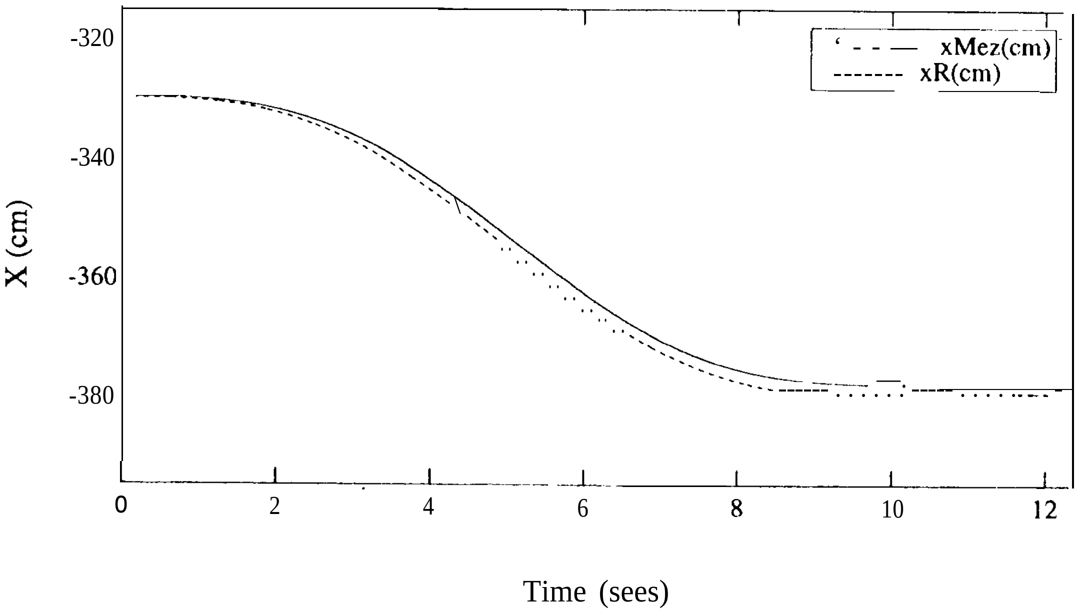


Figure 15c : Xlimit1 Experiment DLS error vs t

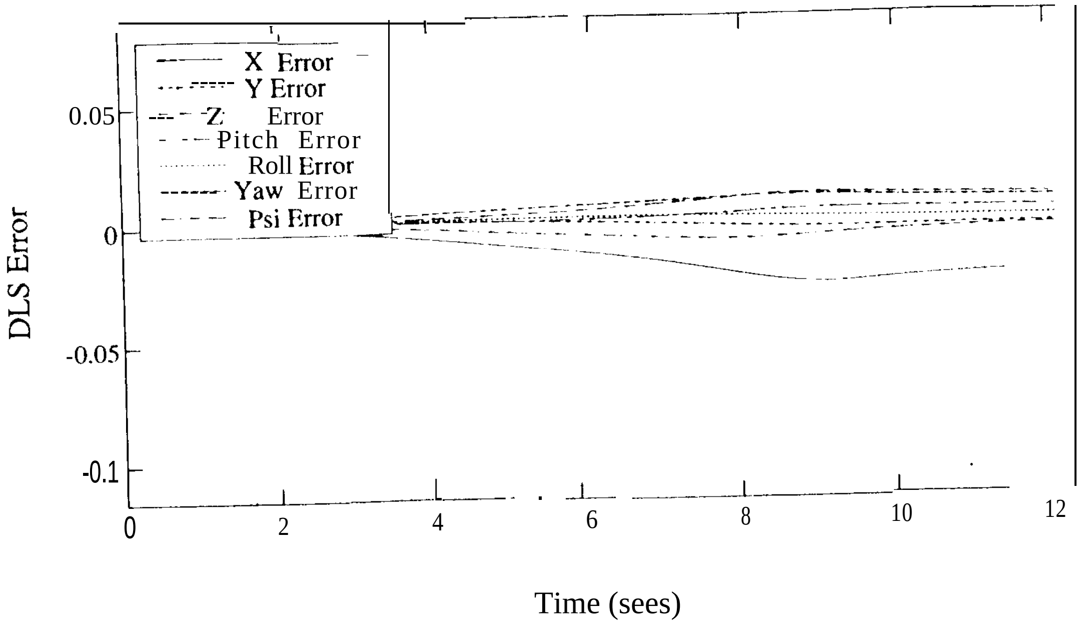


Figure 16a : Xlimit2 Experiment θ_4 vs t

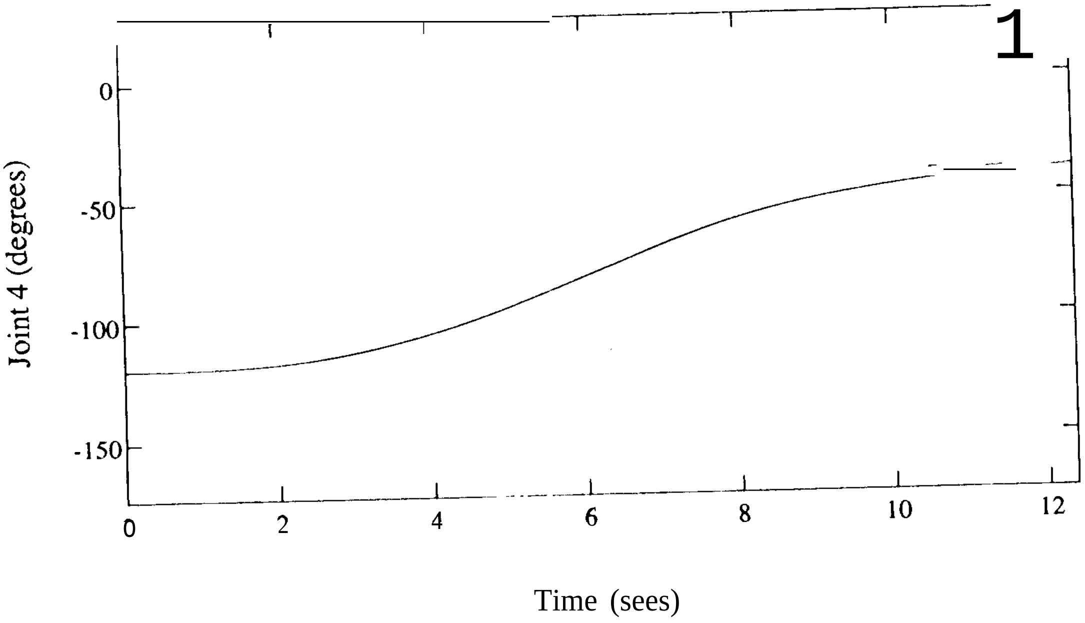


Figure 16b : Xlimit2 Experiment x vs t

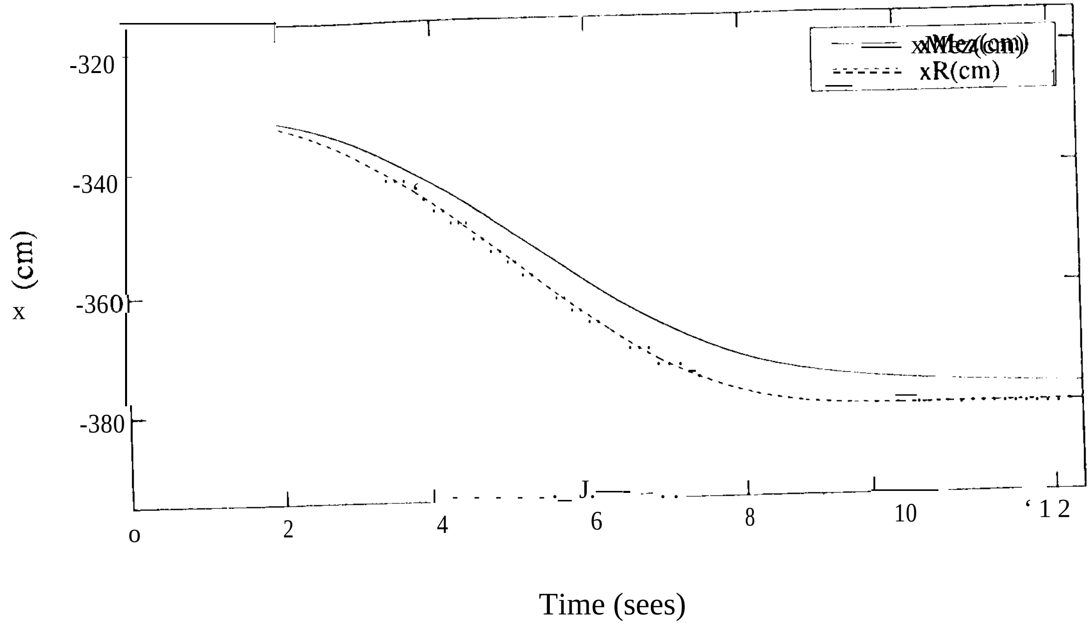
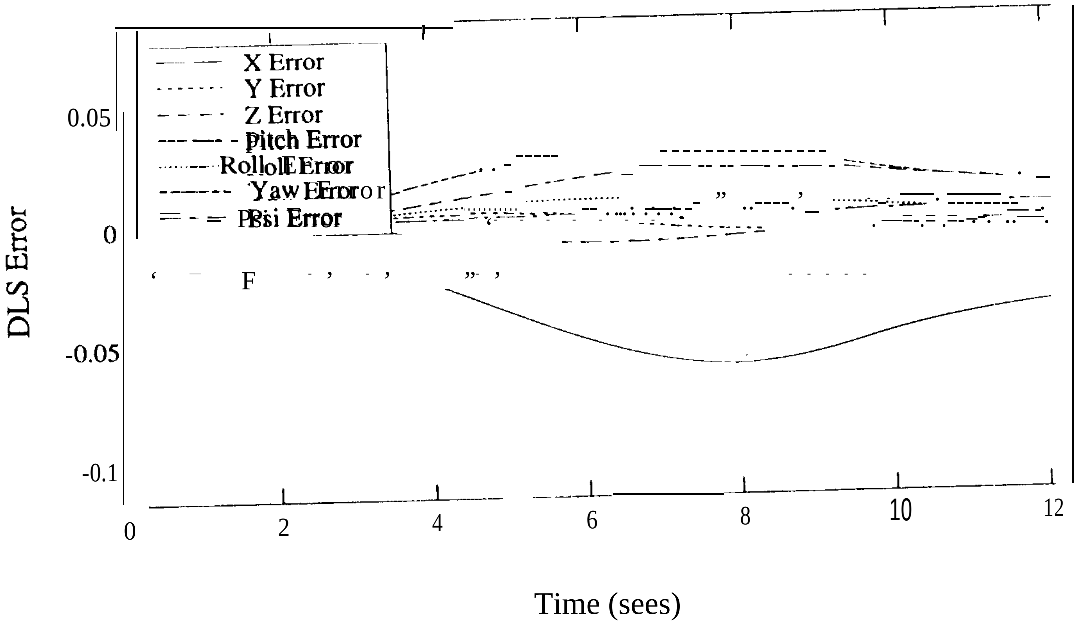


Figure 16C : Xlimit2 Experiment DLS error vs t



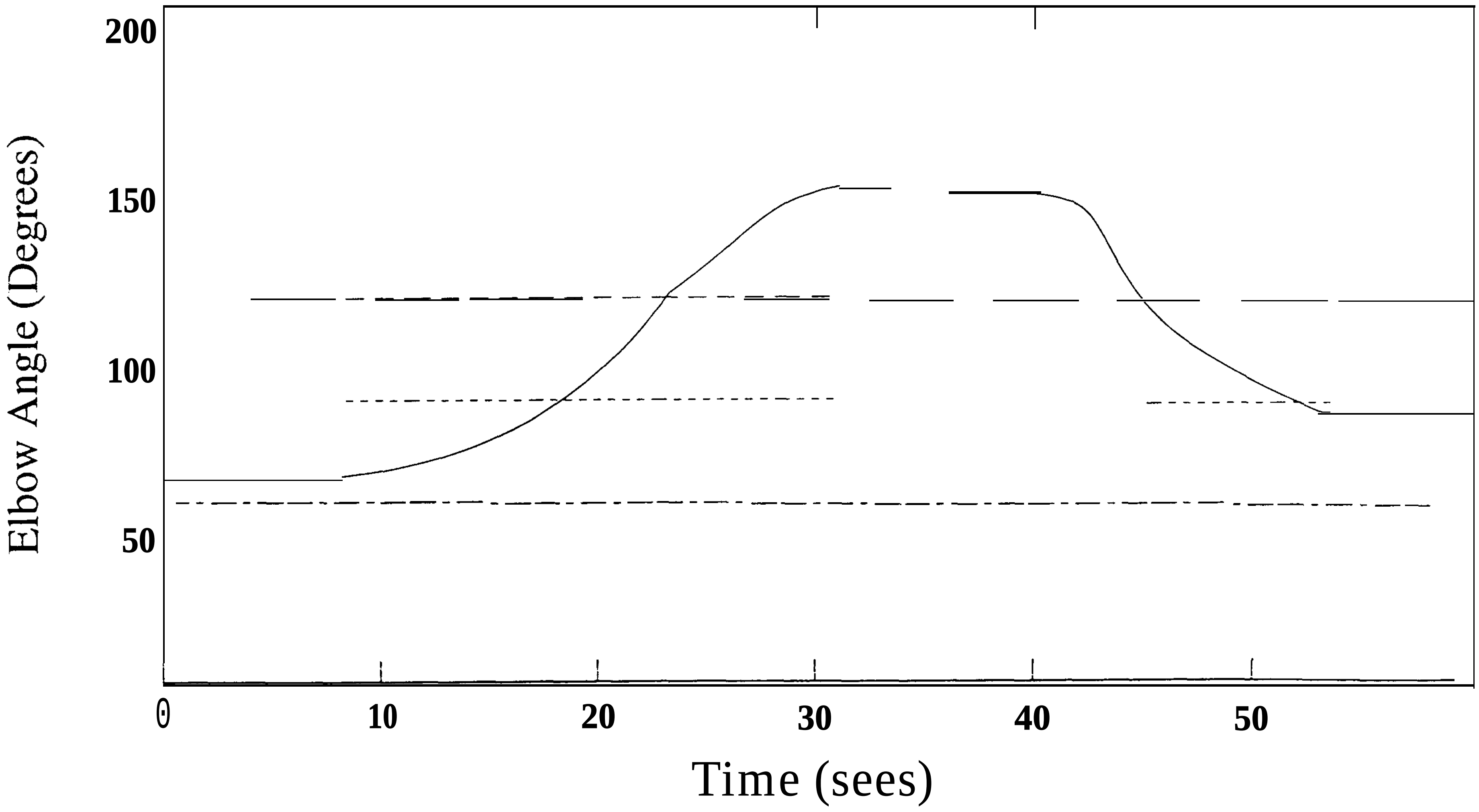


Figure 17: Platform Experiment ϕ vs t